

Helmut Pils

# Das **Linux-Tutorial**

## Ihr Weg zum LPI-Zertifikat

- Linux meistern
- Berufliche Qualifikation verbessern
- LPI-zertifizieren

2. Auflage



Mit Online-Service  
zum Buch

Helmut Pils

**Das Linux Tutorial –  
Ihr Weg zum LPI-Zertifikat**

## Aus dem Bereich IT erfolgreich lernen

### **Grundkurs IT-Berufe**

von Andreas M. Böhm und Bettina Jungkunz

### **Prüfungsvorbereitung für IT-Berufe**

von Manfred Wünsche

### **Grundlegende Algorithmen**

von Volker Heun

### **Grundkurs Programmieren mit Delphi**

von Wolf-Gert Matthäus

### **Grundkurs Visual Basic**

von Sabine Kämper

### **Visual Basic für technische**

### **Anwendungen**

von Jürgen Radel

### **Grundkurs Smalltalk –**

### **Objektorientierung von Anfang an**

von Johannes Brauer

### **Grundkurs Software-Entwicklung mit C++**

von Dietrich May

### **Grundkurs JAVA**

von Dietmar Abts

### **Aufbaukurs JAVA**

von Dietmar Abts

### **Grundkurs Java-Technologien**

von Erwin Merker

### **Objektorientierte**

### **Programmierung in JAVA**

von Otto Rauh

### **Java ist eine Sprache**

von Ulrich Grude

### **Middleware in Java**

von Steffen Heinzl und Markus Mathes

### **Rechnerarchitektur**

von Paul Herrmann

### **Grundkurs Relationale Datenbanken**

von René Steiner

### **Grundkurs Datenbankentwurf**

von Helmut Jarosch

### **Datenbank-Engineering**

von Alfred Moos

### **Grundlagen der Rechnerkommunikation**

von Bernd Schürmann

### **Netze – Protokolle – Spezifikationen**

von Alfred Olbrich

### **Grundkurs Verteilte Systeme**

von Günther Bengel

### **Grundkurs**

### **Mobile Kommunikationssysteme**

von Martin Sauter

### **Grundkurs Wirtschaftsinformatik**

von Dietmar Abts und Wilhelm Müller

### **Grundkurs Theoretische Informatik**

von Gottfried Vossen und Kurt-Ulrich Witt

### **Anwendungsorientierte Wirtschaftsinformatik**

von Paul Alpar, Heinz Lothar Grob, Peter Weimann und Robert Winter

### **Business Intelligence – Grundlagen und praktische Anwendungen**

von Hans-Georg Kemper, Walid Mehanna und Carsten Unger

### **Grundkurs**

### **Geschäftsprozess-Management**

von Andreas Gadatsch

### **Prozessmodellierung mit ARIS®**

von Heinrich Seidlmeier

### **ITIL kompakt und verständlich**

von Alfred Olbrich

### **BWL kompakt und verständlich**

von Notger Carl, Rudolf Fiedler, William Jórasz und Manfred Kiesel

### **Masterkurs IT-Controlling**

von Andreas Gadatsch und Elmar Mayer

### **Masterkurs Computergrafik**

### **und Bildverarbeitung**

von Alfred Nischwitz und Peter Haberäcker

### **Grundkurs Mediengestaltung**

von David Starmann

### **Grundkurs Web-Programmierung**

von Günter Pomaska

### **Web-Programmierung**

von Oral Avci, Ralph Trittman und Werner Mellis

### **Grundkurs MySQL und PHP**

von Martin Pollakowski

### **Grundkurs SAP R/3®**

von André Maassen und Markus Schoenen

### **SAP®-gestütztes Rechnungswesen**

von Andreas Gadatsch und Detlev Frick

### **Kostenträgerrechnung mit SAP R/3®**

von Franz Klenger und Ellen Falk-Kalms

### **Masterkurs Kostenstellenrechnung mit SAP®**

von Franz Klenger und Ellen Falk-Kalms

### **Controlling mit SAP®**

von Gunther Friedl, Christian Hilz und Burkhard Pedell

### **Logistikprozesse mit SAP R/3®**

von Jochen Benz und Markus Höflinger

### **Das Linux-Tutorial – Ihr Weg zum LPI-Zertifikat**

von Helmut Pils

Helmut Pils

# **Das Linux-Tutorial – Ihr Weg zum LPI-Zertifikat**

**Linux meistern – Berufliche  
Qualifikation verbessern –  
LPI-zertifizieren**

Mit 36 Abbildungen

2., durchgesehene und verbesserte Auflage





Bibliografische Information Der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <<http://dnb.ddb.de>> abrufbar.

Das in diesem Werk enthaltene Programm-Material ist mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Der Autor übernimmt infolgedessen keine Verantwortung und wird keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieses Programm-Materials oder Teilen davon entsteht.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne von Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürfen.

Höchste inhaltliche und technische Qualität unserer Produkte ist unser Ziel. Bei der Produktion und Auslieferung unserer Bücher wollen wir die Umwelt schonen: Dieses Buch ist auf säurefreiem und chlorfrei gebleichtem Papier gedruckt. Die Einschweißfolie besteht aus Polyäthylen und damit aus organischen Grundstoffen, die weder bei der Herstellung noch bei der Verbrennung Schadstoffe freisetzen.

1. Auflage April 2004

2., durchgesehene und verbesserte Auflage Dezember 2005

Alle Rechte vorbehalten

© Friedr. Vieweg & Sohn Verlag /GWV Fachverlage GmbH, Wiesbaden 2005

Lektorat: Dr. Reinald Klockenbusch / Andrea Broßler

Der Vieweg-Verlag ist ein Unternehmen von Springer Science+Business Media.

[www.vieweg-it.de](http://www.vieweg-it.de)



Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Verlags unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Konzeption und Layout des Umschlags: Ulrike Weigel, [www.CorporateDesignGroup.de](http://www.CorporateDesignGroup.de)

Umschlagbild: Nina Faber [de.sign](http://de.sign), Wiesbaden

ISBN-13: 978-3-8348-0004-6

e-ISBN-13: 978-3-322-89057-3

DOI: 10.1007/978-3-322-89057-3

---

## Vorwort

---

### ***Willkommen zum Linux Tutorial.***

Ich habe dieses Buch geschrieben, um Ihnen, so weit es in meinen Kräften steht, zu helfen, den eingeschlagenen Weg Linux zu lernen, einzusetzen und mit einem internationalen Industriezertifikat des LPI (Linux Professional Institut) zu krönen, erfolgreich zu meistern.

Das LPI-Zertifikat bildet die Grundlage für die SuSE United Linux-Zertifizierung. Allerdings möchte ich auch nicht den Eindruck erwecken, dass alleine das Studium des Buches Sie in die Lage versetzt, die Prüfung zu bestehen. Ein wichtiger Punkt ist die praktische Arbeit mit dem System.

Ich habe mich bemüht, einen roten Faden durch das Buch laufen zu lassen, so dass der Einsteiger das Buch am besten von vorne nach hinten durcharbeitet. Dabei stand im Vordergrund, den Aufbau so zu gestalten, dass keine Begriffe einfach vom Himmel regnen, sondern die Einführung neuer Bereiche in den Kontext integriert wurden.

Dennoch ist es, wenn diverse Vorkenntnisse vorhanden sind, auch möglich, Teile oder ganze Kapitel zu überspringen, ohne dass man den roten Faden dabei aus der Hand gibt.

Für die geübten Linux-Benutzer unter Ihnen, die einfach eine kompakte Zusammenstellung des Prüfungsstoffes des LPI's suchen, kann dieses Buch auch als Nachschlagewerk für die Prüfungsvorbereitung verwendet werden.

Der Aufbau des Buches folgt dem zu Grunde liegenden didaktischen Konzept, den geneigten Leser an der Hand zu nehmen und ihn vom ersten Kontakt – der ersten Installation – bis hin zu komplexen Netzwerkdiensten zu begleiten. Deshalb beginnt das Buch mit der Planung und Organisation des Installationsprozesses und setzt sich dann fort in der Erforschung der Systemarchitektur.

Nachdem Sie am Ende des Kapitels 3 den Umgang mit dem Standardeditor vi kennen gelernt haben, werde ich Sie in die grundlegende Thematik der Benutzerverwaltung und Festplattenkontingentierung einführen.

---

Kapitel 5 und 6 konzentrieren sich voll auf das Prozesskonzept und den Bootvorgang. Hier lernen Sie, wie man den Bootvorgang steuert und Prozesse gezielt verwaltet.

Die Handhabung neuer Hardware und das Erstellen eines neuen Betriebssystemkernes sind die Inhalte von Kapitel 7 bis 8.

Die Installation und Konfiguration des X11-Servers für die grafische Oberfläche mit den diversen Window-Managern ist Inhalt des Kapitel 9.

Nichts geht ohne Drucker. Deshalb widmet sich Kapitel 10 sehr ausführlich der Installation und Konfiguration von lokalen Druckern und Netzwerkdruckern.

Die Kapitel 11 bis 12 gehen auf die wichtigen Fragen verschiedener Tätigkeiten zur Automatisierung ein. Es werden die notwendigen Komponenten dazu betrachtet.

Das große Thema Netzwerke und Netzwerkdienste werden in den Kapiteln 13 bis 15 einer Klärung zugeführt. Hier werden neben einer allgemeinen Einführung in die Netzwerktechnik vor allem die wichtigen Netzwerkdienste wie Mail-, DNS-, und Web-Server behandelt. Die Kombination von Windows und Linux zu einem heterogenen Netzwerk und der Umgang mit den verschiedenen Freigabesystemen wird in Kapitel 15 genauer erklärt.

Das Kapitel 16 erklärt in kurzer und prägnanter Art die Risiken und sicherheitsrelevanten Themen bei der Arbeit mit einem im Netzwerk befindlichen Computer. Dabei wird anhand eines relativ komplexen Beispiels der Themenbereich Firewall angeschnitten. Für die sichere Fernadministration wird am Ende von Kapitel 16 die Secure-Shell vorgestellt.

In den Anhängen A und B ist zur Überprüfung des Gelernten eine Beispielpfprüfung für das LPI Level I-Zertifikat zusammengestellt. Diese Prüfung gliedert sich in die Teilprüfungen 101 und 102.

Das Buch wird Ihnen helfen, alle Klippen und Stürme auf dem Weg zum LPI Level I-Zertifikat zu umschiffen. Ich wünsche Ihnen beim Studium und Lernen viel Spaß und bei der praktischen Anwendung des Erlernten viel Erfolg.

Melk, im November 2003

Helmut Pils

---

## Inhaltsverzeichnis

---

<b>1</b>	<b>Aller Anfang ist leicht ...</b>	<b>11</b>
1.1	Der Installationsprozess kann beginnen	12
<b>2</b>	<b>Der erste Kontakt</b>	<b>17</b>
2.1	Die Shell	18
<b>3</b>	<b>Ein Blick unter die Motorhaube</b>	<b>29</b>
3.1	Das Dateisystem	29
3.1.1	Das ext2 Filesystem	31
3.1.2	ext3 – oder wie wird ein Journal geführt	34
3.2	Die Systemarchitektur	35
3.2.1	Partitionieren und Ordnung schaffen	38
3.2.2	Die wichtigsten Befehle	46
3.2.3	Der vi – alt aber gut	53
3.2.4	Mitgefangen – Mitgehangen	60
<b>4</b>	<b>Die Benutzerverwaltung</b>	<b>67</b>
4.1	Der Benutzer	67
4.2	Nicht jeder darf alles	77
4.3	Die Quote	86
4.4	Hilfe ist immer und überall	89
<b>5</b>	<b>Prozesse, Prozesse</b>	<b>95</b>
5.1	Prozesse – oder wie behalte ich den Überblick	95
5.2	Befehle, Befehle oder ein Prozess kommt selten allein	107
5.3	Texte bearbeiten – Jetzt beginnt der Spaß	115
5.4	Eine kleine Kanalarundfahrt	127
5.5	Und noch mehr Befehle	134
<b>6</b>	<b>Zurück an den Start</b>	<b>143</b>
6.1	Alle ins Boot – der Bootmanager	143

---

6.2	Runlevel – bitte nicht davonlaufen .....	149
6.3	Big Brother – Alles wird protokolliert.....	157
<b>7</b>	<b>Neues ist immer Willkommen.....</b>	<b>169</b>
7.1	Das Herz des Systems.....	169
7.2	Neue Software braucht das Land .....	181
7.3	Aus der Quelle schöpfen.....	185
7.4	Alle wollen Red Hat-Pakete.....	192
7.5	Etwas zum Genießen – Die Debian-Paketverwaltung.....	197
<b>8</b>	<b>Jedermanns Freude – Neue Hardware .....</b>	<b>207</b>
8.1	Am Anfang war das BIOS.....	207
8.2	Hardware und das System.....	208
<b>9</b>	<b>X11 und die Welt wird bunt.....</b>	<b>223</b>
9.1	Der Grafikzauber .....	223
<b>10</b>	<b>Drucken – alles muss zu Papier .....</b>	<b>247</b>
10.1	Der Herr der schwarzen Zunft – Der Druckerdaemon lpd.....	247
10.2	Filter über Filter aber kein kalter Kaffee .....	250
10.3	Ohne Manager geht gar nichts .....	253
10.4	Nun bringen wir's zu Papier .....	255
10.5	Auch die Schlange will verwaltet werden.....	259
<b>11</b>	<b>Shell-Skripts braucht das Land.....</b>	<b>263</b>
11.1	Eine kleine Retrospektive .....	263
11.2	Hello World.....	266
11.3	Der Werkzeugkoffer .....	267
11.4	Ein kleiner Shell-Nachschlag .....	281
<b>12</b>	<b>Automatisch oder Bequemlichkeit über alles .....</b>	<b>283</b>
12.1	Auf den richtigen Zeitpunkt kommt es an.....	283
12.2	Ein Muss - Sicherheitskopien.....	290
<b>13</b>	<b>Ohne Netz geht nichts.....</b>	<b>303</b>
13.1	Eine kurze Geschichte .....	303
13.2	Schichten und Netzwerktypen.....	304
13.3	Protokolle, Adressen und Ports .....	311

---

13.4	Reine Einstellungssache .....	318
13.5	Nützliche Netzwerkdienste .....	333
<b>14</b>	<b>Server und andere dienstbare Geister .....</b>	<b>355</b>
14.1	Der DNS-Server .....	355
14.2	Mail-Server – Kommunikation ist alles .....	363
14.3	Die große weite Welt – der Web-Server .....	371
<b>15</b>	<b>Freigaben oder die Würze des Netzwerks .....</b>	<b>377</b>
15.1	Der NFS-Dienst .....	377
15.2	Samba – das Fenster zu Windows.....	383
<b>16</b>	<b>Auch Sicherheit muss geprüft werden .....</b>	<b>393</b>
16.1	Sicherheitsrichtlinien – was sollte geprüft werden .....	393
16.2	Walle, Walle - Firewall.....	401
16.3	Die Secure-Shell oder alles ist verschlüsselt .....	411
<b>A</b>	<b>Testfragen für Prüfung 101.....</b>	<b>417</b>
<b>B</b>	<b>Testfragen zur Prüfung 102 .....</b>	<b>449</b>
	<b>Schlagwortverzeichnis .....</b>	<b>479</b>

---

## Aller Anfang ist leicht ...

---

... oder wie kommt Linux auf meinen Computer?

Bevor man noch richtig loslegen kann, begegnet man schon dem ersten großen Problem – welches Linux nehme ich?

Das amerikanische Red Hat oder Debian, das deutsche SuSE, das französische Mandrake oder, oder, oder, .... Bei derzeit ca. 96 unterschiedlichen Distributionen wahrlich keine leichte Aufgabe, das Passende zu finden.

### *Hinweis*

Es werden immer wieder die Begriffe Distribution und Linux synonym verwendet. Ich möchte hier die Gelegenheit ergreifen und dies etwas näher beleuchten. Unter dem Begriff Linux ist lediglich der Kernel zu verstehen. SuSE, Red Hat, Debian, ... und wie sie alle heißen, sind Distributionen. Diese sammeln rund um den Linux-Kernel noch eine Unzahl von mehr oder weniger nützlichen Tools und Programme, sie entwickeln eigene Installationsroutinen, die immer benutzerfreundlicher werden und somit den Nimbus des schwer zu installierenden Betriebssystems zusehends Lügen strafen.

Ich möchte in diesem Buch distributionsunabhängig bleiben. Dennoch werde ich immer wieder die Unterschiede der drei wichtigsten Linux-Distributionen aufzeigen und besprechen. Da Red Hat im angloamerikanischen Raum der Marktführer ist und SuSE im europäischen Raum dominiert, werden diese beiden Distributionen behandelt. Für Hardcore-Fans kommt natürlich auch Debian, welches zu Recht den Ruf hat, das stabilste System zu sein, vor.

Damit habe ich die Auswahl der richtigen Distribution schon vorweg genommen. Wir starten damit, dass wir versuchen, die aktuelle Red Hat-Distribution auf einen Rechner zu installieren. Wir gehen dabei noch nicht auf das Warum ein – das kommt später noch. Wir wollen lediglich ein lauffähiges System, damit wir damit arbeiten können. Eine sogenannte Dualbootvariante – „Windows und Linux“ parallel auf einer Platte – steht im Kapitel 12.2. beschrieben.

## 1.1

### Der Installationsprozess kann beginnen

Wir haben einen freien PC, der ausschließlich für Linux verwendet werden soll. Es sind also keinerlei Daten auf dem PC, die nicht gelöscht oder überschrieben werden dürfen.

Ferner halten wir die Red Hat-Distribution in Händen und erkennen, dass neben den 3 Installations CDs auch noch eine DVD in dem Paket vorhanden ist. Wir können bei Vorhandensein eines DVD-Laufwerkes auch die DVD zur Installation verwenden. Der Installationsprozess unterscheidet sich nur darin, dass man bei den CDs aufgefordert wird, die jeweils nächste CD einzulegen – also Disc-Jockey zu spielen, wohingegen bei der DVD der Prozess ohne Discwechsel abläuft. Ich werde den DVD-Weg beschreiten, da mein Computer ein DVD-Laufwerk eingebaut hat, und außerdem kann ich während des Installationsvorganges einen Kaffee trinken gehen.

#### Hinweis

Während ich an diesem Buch geschrieben habe, hat Red Hat bekannt gegeben, dass ihre nächsten Distributionen ausschließlich Online via Download vertrieben werden sollen.

Die DVD ist ebenso bootfähig wie die erste CD. Damit der Computer von CD oder DVD booten kann, muss man im BIOS diese Option erst einstellen (Abbildung 1). Da es verschiedene Hersteller von BIOS'es gibt, die sich nicht so sehr in der grundsätzlichen Funktionalität unterscheiden, sondern vielmehr durch die Menüdarstellung, werde ich nur exemplarisch auf das mir zur Verfügung stehende eingehen. Die hier und in Folge benötigten Einstellungen finden sich in ähnlicher Weise auf jedem andern BIOS wieder.

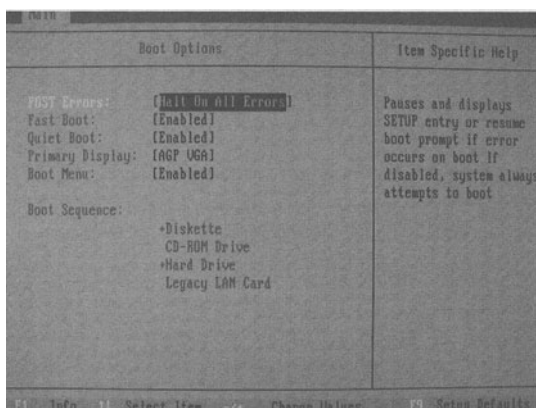


Abbildung 1: BIOS – Boot from CD



Wir haben nun das BIOS so eingestellt, dass wir den Installationsprozess von CD/DVD starten können. Wir legen die DVD ein und starten den Computer neu. Das Startmenü bestätigen wir einfach mit der Enter-Taste (Return). Daraufhin wird der Linux Kernel in den Speicher geladen und gestartet. Nachdem wir die Sprachauswahl auf Deutsch umgestellt haben, können wir das zu benutzende Keyboard auswählen. Wir verwenden den Eintrag **German (latin 1 /w no deadkeys)**. Das heißt, wir verwenden eine Tastatur ohne sogenannte Metatasten (die Windows-Taste wäre eine solche Metataste). Wir haben bereits erkannt, dass das Installationsprogramm so weit entwickelt ist, dass eigentlich alle vorgeschlagenen Auswahlpunkte übernommen werden können.

Beim folgenden Installationstyp wählen wir Workstation und danach das automatische Partitionieren.

*Achtung*

Alle Daten, die zuvor auf dem PC vorhanden waren, werden vollständig gelöscht!!

Wir wählen die Option: **Alle Partitionen auf dem System entfernen**. Damit wird eine von Red Hat vordefinierte Festplattenaufteilung realisiert.

Jedes Betriebssystem benötigt ein spezielles Programm, welches dazu da ist, vom BIOS aufgerufen zu werden und das eigentliche Betriebssystem zu starten. Der sogenannte Bootloader. Wir bestätigen mit Weiter den Vorschlag für den Bootloader genauso wie die Netzwerkeinstellungen. Danach kommt ein Bildschirm, der es uns gestattet, eine Desktop Firewall zu installieren. Damit wir uns beim Einarbeiten in das System nicht selbst behindern, wählen wir zu diesem Zeitpunkt den Sicherheitslevel **Keine Firewall** aus.

Bei der Zeitzoneangabe suchen wir die jeweilige Zeitzone, z. B. wenn wir in Wien zu Hause sind, **Europa/Wien** aus. Danach wird ein Passwort für den „Chef“ des Systems abgefragt. Der Benutzer, der alle, und ich meine hier wirklich alle Rechte auf dem System besitzt, heißt **root**. Das Passwort sollte also dementsprechend sicher sein.

Danach erhalte ich noch die Möglichkeit, zusätzliche zu installierende Software auszuwählen, was wir allerdings nicht tun. Noch zweimal den Bildschirm bestätigen, und erst jetzt beginnt der eigentliche Installationsprozess. Die Dauer der Installation richtet sich nach der Leistungsfähigkeit der verwendeten Hardware.

Jetzt ist ein guter Zeitpunkt, einen Kaffee zu genießen.

*Nach 30 min.*

Zurück vom Kaffee, kann es mit Riesenschritten weiter gehen. Die Installation neigt sich dem Ende zu. Wir bestätigen, dass wir keine Bootdiskette benötigen.

Danach kommt die Auswahl der Grafikkarte und des Monitors. Auch diese Geräte werden normalerweise richtig erkannt. All denjenigen, bei denen die Erkennung fehl geschlagen ist, sei ein kurzer Trost gesendet. Entweder Sie wissen den Typ und stellen ihn manuell ein oder sie kreuzen die Option **X-Server Einstellung überspringen** an und gedulden sich, bis wir das nötige Know How für das Troubleshooting haben. Danach wird die CD/DVD automatisch ausgeworfen, und mit einem Klick auf Beenden wird der PC neu gestartet.

Es sind noch ein paar Abschlussarbeiten notwendig, bevor wir uns an das Kennenlernen des Systems machen können. Nachdem der root-User (mit dem Administrator auf Windows zu vergleichen) oder auch **SuperUser** genannt, sehr mächtig ist, sollte man unter normalen Umständen seine Alltagsarbeit nicht unter diesem Account verrichten. Deshalb benötigen wir einen weiteren, nicht privilegierten Benutzer auf dem Computer, den wir nun anlegen wollen.

Wir beantworten einfach die Felder, die uns angezeigt werden, mit einem Account-Namen, z. B. **helmut** und einem Passwort.

*Hinweis*

Bitte beachten Sie, dass ein Linux-System **case sensitiv** ist, also zwischen Groß- und Kleinschreibung unterscheidet. Sie sollten sich bei der Arbeit mit einem Linux-System angewöhnen, alles klein und ohne Zwischenräume zu schreiben. Sie machen sich damit das Leben einfacher – wie Sie später noch sehen werden.

Nun kann noch einmal die Systemzeit überprüft und gegebenenfalls geändert werden. Falls Sie eine Soundkarte installiert haben, wird diese selbst erkannt und kann getestet werden. Gibt es zu diesem Zeitpunkt noch Probleme mit dem Sound, dann gedulden Sie sich noch ein wenig, wir werden bald alle Aspekte näher kennen lernen. Wir benötigen keine weiteren Einstellungen und bestätigen die restlichen Fenster ungeändert. Anschließend kommt der grafische Anmeldebildschirm. Bei diesem Anmeldebildschirm melden Sie sich bitte mit dem Benutzer root und dem zugehörigen Passwort an. Sagte ich oben, dass mit root arbeiten gefährlich sei und deshalb nicht zu Routinearbeiten verwendet werden sollte? Stimmt – aber wir haben ja keine Routineaufgaben vor uns, sondern wir wollen echt gute Systemadministratoren werden.

Nach dem Anmelden wird ein sogenannter Desktop-Manager oder Window-Manager geladen, der das X-Window-System verwendet.

Wir haben jetzt ein lauffähiges Linux Betriebssystem auf unserem Computer installiert. Viele von den Einstellungen und deren Auswirkungen, die wir bei der Installation vorgenommen haben, sind uns noch unbekannt. Es wird höchste Zeit, sich mit dem Betriebssystem und seinen Eigenheiten näher auseinander zu setzen.

---

# 2

## Der erste Kontakt

---

In diesem Kapitel werden wir uns ganz vorsichtig an das Linux-Betriebssystem herantasten. Wir haben zwar eine grafische Oberfläche zur Verfügung, aber wir werden trotzdem mit einem zeichenbasierenden Tool, dem Terminal, das ist eine Art Kommunikationstunnel zwischen Benutzer und dem Betriebssystemkern, arbeiten.

Man kann das Terminal mit der DOS-Box unter Windows vergleichen. In dem Terminal wird ein Programm gestartet, welches **Shell** heißt. Der Name Shell kommt daher, dass dieses Programm den Betriebssystemkern wie eine Muschel umschließt. Die prinzipielle Architektur eines Linux Systems kann man sich so wie in Abbildung 2 vorstellen.

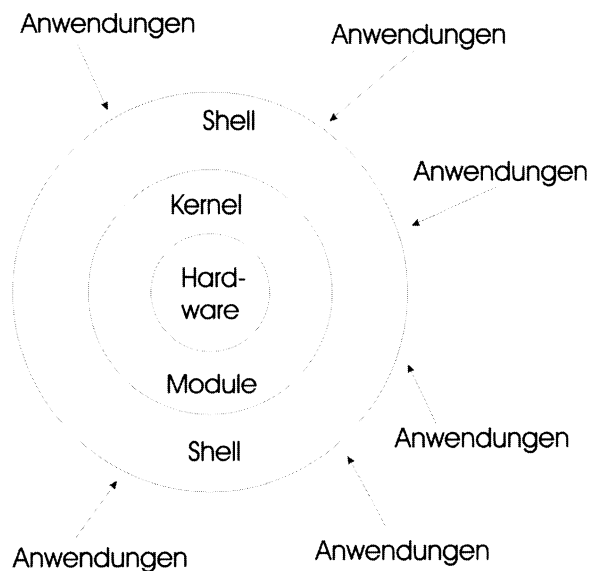


Abbildung 2: Linux Systemarchitektur

## 2.1

### Die Shell

Die Shell ist ein Vermittler zwischen den Welten. Zwischen der Welt des Benutzers und der des Systemkerns. Es gibt sehr viele unterschiedliche Arten von Shells. In Tabelle 1 sind nur die, nach Meinung des Autors, wichtigsten Vertreter von Shells aufgelistet, ohne Anspruch auf Vollständigkeit zu erheben.

**Tabelle 1: Linux Shells**

<i><b>Shell</b></i>	<i><b>Erklärung</b></i>
bash	Die Bourne Again-Shell ist die eigentliche Standard-Shell von Linux.
sh	Die Bourne-Shell ist die älteste Shell, die Linux kennt, und ist oftmals ein Verweis auf die bash.
csh	Das ist die Berkley Unix C-Shell. Der Name kommt daher, dass hier einige Anleihen von der Programmiersprache C genommen wurden.
tcsh	Das ist eine erweiterte Version der csh.
ksh	Die Korn-Shell. Das ist eine kommerzielle Shell und ist die Standard-Shell bei SUN Solaris. Auf der Linux-Seite gibt es eine freie Portierung der Korn-Shell, die pdksh (Public Domain Korn Shell)

Wir werden durchwegs mit der bash arbeiten. Auf der Interaktionsebene der sogenannten Kommandozeile werden die Befehle eingetippt, von der Shell interpretiert und an den Kernel weitergegeben. Dieser verarbeitet die Befehle und gibt sein Resultat auf der benutzten Oberfläche, im Terminal, wieder aus.

Damit wir die Shell benutzen können, müssen wir ein wenig Vorarbeit leisten. Wir klicken auf den roten Hut links unten in der Ecke. In dem erscheinenden Popup-Fenster gehen wir mit der Maus auf den Eintrag Systemtools und darin auf den Eintrag Terminal. Nun klicken wir auf den Eintrag Terminal und halten die Maustaste gedrückt. Wir ziehen den Eintrag (das **Icon**) auf den Bildschirmhintergrund – den **Desktop**. Ein Doppelklick öff-

net das Terminal-Programm und damit die Shell. Die Shell ist das wichtigste Interaktions-Tool, das es auf einem Linux-System gibt. Im Terminal-Fenster erhalten wir einen **Prompt**.

```
[root@localhost root]#
```

Der Prompt ist eine Eingabeaufforderung. Dort können wir Befehle eingeben und Programme starten.

Der Prompt `root@localhost root]#` bedeutet: Benutzer (`root`) auf (`@`) dem Rechner (`localhost`) im Verzeichnis (`root`) und einige Abschlusszeichen (`]#`). Wobei das schließende Raute-Zeichen immer den SuperUser, also den Benutzer `root`, kennzeichnet. Dies entspricht einer nicht verpflichtenden Konvention, damit der SuperUser auch optisch gekennzeichnet wird.

Nicht umsonst wird Linux ob seiner freien Konfigurierbarkeit gelobt. Wir können das Erscheinungsbild des Prompts so einstellen, dass er unseren Vorstellungen ganz und gar entspricht. Das Aussehen des Prompts wird in sogenannten **Umgebungsvariablen** definiert. Was sind diese Umgebungsvariablen? Wir befinden uns ja im Terminal-Programm, in dem die Shell (welche die Befehle interpretiert) läuft. Das Verhalten und die Eigenschaften der Shell, also die Umgebung, kann man einstellen und zwar in den Umgebungsvariablen oder Shell-Variablen.

Sehen wir uns die Umgebungsvariablen, die derzeit schon definiert sind, einmal an. Dazu geben wir auf der Kommandozeile den Befehl **env** ein und bestätigen ihn mit Return.

Mit dem Befehl `env` erhält man in der Regel als Ausgabe eine Untermenge von dem, was das Kommando **set** liefert.

```
[root@localhost root]#set
```

Als Ausgabe erhalten wir (Auszug):

```
BASH=/bin/bash
BASH_ENV=/root/.bashrc
COLORS=/etc/DIR_COLORS.xterm
COLORTERM=gnome-terminal
COLUMNS=80
DISPLAY=:0.0
```

```
G_BROKEN_FILENAMES=1
HISTFILE=/root/.bash_history
HISTFILESIZE=1000
HISTSZ=1000
HOME=/root
HOSTNAME=localhost.localdomain
MAIL=/var/spool/mail/root
OLDPWD=/root
OPTERR=1
OPTIND=1
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin
:/usr/bin:/usr/X11R6/bin:/root/bin
PS1='[\u@\h \w]\$ '
PS2='> '
PS4='+ '
PWD=/root/.vnc
SHELL=/bin/bash
TERM=xterm
USER=root
. . . .
```

Wir bekommen eine Liste von allen lokal gesetzten Variablen. Darunter sind auch die (hervorgehobenen) Variablen, die das Aussehen und das Verhalten des Prompts bestimmen.

Um einzelne Variablen ausgeben zu können, verwenden wir das **echo**-Kommando. Um nur den Inhalt der Variablen **PS1** zu erhalten, gibt man folgenden Befehl ein:

```
[root@localhost root]#echo $PS1
[\u@\h \w]\$
[root@localhost root]#
```

Das \$ vor der Variablen kann man als Variablen-Inhaltsoperator begreifen. Zum Setzen von Variablen benötigt man das \$ nicht, aber um den Inhalt einer Variablen anzeigen zu lassen, ist das \$ notwendig.

Die Variablen PS1, PS2, PS3 und PS4 müssen nicht alle gesetzt sein und haben folgende Bedeutung:

**Tabelle 2: Umgebungsvariablen für den Prompt**

<b>Variable</b>	<b>Bedeutung</b>
PS1	Definiert den Primary Prompt String – Dies wird also als Prompt auf der Kommandozeile ausgegeben.
PS2	Wenn eine Eingabezeile nicht vollständig eingegeben worden ist, verwendet die Shell das Zeichen, das hier definiert wurde.
PS3	Die Zeichen in dieser Variablen werden von Tool select verwendet.
PS4	Wenn die Option -x beim set Befehl angegeben wird, kommt diese Variable zum Einsatz und wird vor jedem Befehl ausgegeben.

Für uns am interessantesten sind die zugeordneten Werte der Variablen PS1. Grundsätzlich kann man alle Zeichen des Zeichensatzes verwenden, doch gibt es einige Konstrukte, die besondere Bedeutung haben wie z. B.:

**Tabelle 3: Sonderzeichen**

<b>Wert</b>	<b>Bedeutung</b>
\u	Username des aktuellen Benutzers.
\h	Computernamen, auf dem die Shell ausgeführt wird.
\w	Aktuelles Arbeitsverzeichnis.
\d	Aktuelles Datum.
\t	Aktuelle Zeit.
\n	Neue Zeile.



\\	Backslash.
\[ \]	Definiert eine Serie von nichtdruckbaren Zeichen.
\xxx	Das korrespondierende Zeichen zum oktalen Wert xxx.
\$(Befehl)	Ausgabe des Kommandos Befehl – Der Befehl sollte allerdings sinnvoll sein!

*Beispiel:*

Experimentieren Sie mit der Variablen PS1 und lassen Sie jeweils die geänderten Inhalte mit dem echo-Befehl ausgeben.

```
PS1=$(date)
```

```
PS1=$(ls -l)
```

```
PS1="\u"
```

```
PS1="\h"
```

```
PS1= "\h $"
```

Mit dem echo-Befehl kann man auch beliebige Zeichen am Bildschirm ausgeben z. B.:

```
[root@localhost root]#echo Hallo
```

```
Hallo
```

```
[root@localhost root]#
```

Wenn wir nun beabsichtigen, dass in einer Variablen – wir wollen sie erste nennen – das Wort Hallo stehen soll, müssen wir selbst eine Variable setzen, und das funktioniert folgendermaßen:

```
[root@localhost root]#erste='Hallo'
```

```
[root@localhost root]#echo $erste
```

```
Hallo
```

```
[root@localhost root]#
```

Damit haben wir eine sogenannte lokale Variable erstellt. Wir haben oben von Umgebungsvariablen gesprochen, und was soll jetzt bitte die lokale Variable sein?

Wenn man aus der Shell heraus ein Programm startet, erbt dieses Programm die Umgebung, also alle Umgebungsvariablen der Shell. Alle lokalen Variablen der Shell kennt dieses Programm nicht. Um eine lokale Variable zu einer Umgebungsvariablen zu machen, muss man sie **exportieren**.

```
[root@localhost root]#export erste
```

Nachdem wir die Variable exportieren und nicht nur auf den Inhalt zugreifen wollen, können wir hier das \$ vor dem Variablennamen erste weglassen, denn das \$ brauchen wir nur, wenn wir auf den Inhalt der Variablen zugreifen wollen.

Nun gibt es ein paar Variablen, die man als Systemvariablen begreifen kann und die spezielle Bedeutung haben. Auch wenn die Erklärung vielleicht etwas mystisch klingen mag, soll dies doch hier der Vollständigkeit halber aufgeführt werden. Es handelt sich dabei um die Variablen:

**Tabelle 4: Systemvariablen**

<b>Variable</b>	<b>Bedeutung</b>
\$	Die Variable \$ zeigt die aktuelle Prozess-ID der Shell. Auf den Inhalt kann mit echo \$\$ zugegriffen werden.
?	Zeigt das Ergebnis (Erfolg/Misserfolg) des zuletzt ausgeführten Kommandos an. Auf den Inhalt kann mit echo \$? zugegriffen werden. (Mögliche Werte sind 1 und 0)
!	Zeigt die Prozess-ID des letzten Kindprozesses. Mit echo !\$ kann wieder auf den Inhalt zugegriffen werden.

Wir können somit lokale Variablen anlegen und zu Umgebungsvariablen machen. Wir sind auch in der Lage, den Inhalt von Variablen anzuzeigen und mit `env` (Umgebungsvariablen) und `set` (lokale Variablen) eine Auflistung zu erstellen.

### *Einschub*

Wir machen hier einen kleinen Einschub, obwohl eigentlich zum Themenkomplex Dateisystem gehörend, betrachten wir ein paar der am häufigsten gebrauchten Befehle, die man auf der Kommandozeilebene benötigt.

Eines der wichtigsten, wenn nicht das wichtigste Kommando überhaupt ist **ls** (list). Damit kann man sich den Inhalt des aktuellen Verzeichnisses ausgeben lassen. Hier gleich eine Besonderheit!

### *Achtung*

Bei Linux-Systemen werden Pfadangaben mit `/` und nicht mit `\` so wie unter Windows-Systemen geschrieben. Außerdem gibt es unter Linux das Konzept der Laufwerksbuchstaben nicht!

```
[root@localhost root]#ls
anaconda-ks.log  install.log  install.log.syslog
[root@localhost root]#
```

Diese drei Dateien sind in dem Verzeichnis, in dem ich mich gerade befinde. Dies ist übrigens das persönliche Heimatverzeichnis des SuperUsers `root`.

Um in ein Verzeichnis wechseln zu können, verwendet man den Befehl **cd** (change directory).

```
[root@localhost root]#cd /tmp
[root@localhost tmp]#
```

Damit haben wir das Verzeichnis gewechselt und befinden uns (der Prompt zeigt es auch an) im Verzeichnis `tmp`. Dies ist ein für alle zugängliches und verwendbares Verzeichnis. Um wieder in das private HOME-Verzeichnis des SuperUsers zu wechseln, können wir entweder einfach `cd` und Return eingeben oder im langen Format mit `cd /root`.

Dies soll aber genug des Vorgriffes sein, und wir wenden uns wieder der Shell zu.

Einige von Ihnen werden sich schon gefragt haben, warum ich bei den obigen Beispielen einmal `"` (doppelte Hochkommata) und einmal `'` (einfache Hochkommata) verwendet habe.

Die Hochkommata oder auch **quoting** genannt ist vielleicht das am schwierigsten zu verstehende Konzept eines Linux-Systems.

Generell gibt es drei Arten von Hochkommata:

**Tabelle 5: Hochkommata**

<b>Art</b>	<b>Bedeutung</b>
" "	Hebt die Bedeutung aller Zeichen auf ausgenommen von \, ', \$.
, ,	Hebt die Bedeutung aller Zeichen auf ausgenommen von `.
`cmd`	Substituiert das Kommando cmd mit dem Wert des Kommandos.

Was soll das heißen? Um es besser verstehen zu können, benötigen wir den Begriff der **Metacharakter**. Metacharakter oder auch **Wildcards** genannt, haben im betreffenden Kontext eine besondere Bedeutung.

Wenn ich z. B. alle Dateien in einem Verzeichnis betrachten möchte, die mit jpg im Dateinamen aufhören, kann ich folgenden Befehl verwenden:

```
[root@localhost root]#ls *.jpg  
herald.jpg susi.jpg franz.jpg
```

Das heißt, in diesem Kontext hat das Zeichen \* eine besondere Bedeutung, und zwar ist es ein Wildcard und bedeutet beliebig viele beliebige Zeichen.

Sie haben es schon erraten, es gibt noch weitere Metacharakter. Wir kennen schon einige, die wir bereits verwendet haben, das \$ für die Variablen und natürlich die oben angesprochenen Hochkommata. Zusätzlich existieren noch:

**Tabelle 6: Metacharakter**

<b>Metacharakter</b>	<b>Bedeutung</b>
<b>*</b>	Beliebig viele, beliebige Zeichen
<b>;</b>	Kommando-Trenner
<b>?</b>	Ein, aber ein beliebiges Zeichen
<b>[ ]</b>	Ersetzt eines der in den Klammern aufgeführten Zeichen mit entweder oder.
<b>\</b>	Hebt die Bedeutung des nachfolgenden Zeichens auf und behandelt es als Literal.
<b> </b>	Pipe-Zeichen. Sendet die Ausgabe eines Kommandos zur Eingabe des nachfolgenden Kommandos.
<b>&lt;, &lt;&lt;, &gt;, &gt;&gt;</b>	Dateiumleitungszeichen

Will ich – über den Sinn kann man hier sicherlich diskutieren – eine Datei auflisten, die den Namen **\*** besitzt, dann komme ich mit dem Kommando

```
[root@localhost root]#ls *
```

nicht sehr weit, denn das Zeichen **\*** hat ja die Bedeutung, beliebig viele beliebige Zeichen auszugeben, und ich bekomme alle Dateien in diesem Verzeichnis angezeigt. Wir müssen die Bedeutung des Metazeichens **\*** aufheben. Ein Blick auf die Tabelle zeigt uns, dass dies wieder mit einem Metazeichen funktioniert, und zwar mit dem **\**.

```
[root@localhost root]#ls \  
*
```

Damit haben wir unser Ziel erreicht. Versuchen Sie das Ergebnis der folgenden Befehle zu interpretieren:

```
echo `date`  
echo `date`  
echo "$date"
```

Das Kommando `date` gibt uns das aktuelle Datum und die Uhrzeit aus. Was passiert, wenn man den Befehl `date` eingibt?

Die Shell liest die Umgebungsvariable ***PATH*** aus. In der ***PATH***-Variable sind verschiedene Pfadangaben (das sind Verzeichnisse), mit jeweils einem `:` getrennt gespeichert. Nun versucht die Shell, das eingegebene Kommando zu finden und zwar im ersten Verzeichnis, welches in der ***PATH***-Variablen gespeichert ist. Wenn es dort nicht vorhanden ist, wird im zweiten Verzeichnis gesucht und so weiter. Sobald das Kommando gefunden wurde, wird es ausgeführt. Wird es nicht gefunden, dann gibt die Shell die Meldung

```
bash: Kommando: command not found
```

aus. Wenn man ein neues Verzeichnis in den Suchpfad mit aufnehmen möchte, funktioniert dies so:

```
[root@localhost root]#export PATH=$PATH:/tmp
```

Mit dieser Anweisung fügen wir unserem derzeitigen Pfad das Verzeichnis `/tmp` an der letzten Position hinzu. Überprüfen Sie die Korrektheit, indem Sie sich die Umgebungsvariable ***PATH*** ausgeben lassen.

#### *Einschub*

Für Umgebungsvariablen gilt die Konvention, dass diese Variablen in Großbuchstaben geschrieben werden.

#### *Hinweis*

Alle Änderungen, die wir bisher getätigt haben, gehen verloren, wenn man das Terminal schließt und ein neues Terminal (und damit Shell) startet.

Natürlich gibt es Möglichkeiten, diese Änderungen dauerhaft vorzunehmen. Es gibt verschiedene Dateien, mit denen man die Voreinstellungen, wenn die Shell gestartet wird, definieren kann. Dies sind die Dateien ***/etc/profile*** und ***.bash\_rc***.

Die Datei `/etc/profile` beinhaltet alle Shellvorgaben für alle User des Systems. Diese kann nur der Systemadministrator, also der SuperUser `root`, verändern. Das sind systemweite, userübergreifende Einstellungen. In der Datei `.bash_rc` sind jene Ein-

stellungen vorzunehmen, die pro User zutreffen sollen. Der User selbst kann dort auch Einstellungen vornehmen.

Um das Thema Shell abzuschließen, wollen wir einen Blick auf das Kommando **chsh** (change shell) richten. Mit diesem Kommando kann man dauerhaft, also über das Abmelden und neuerliche Anmelden am System, die Standard-Shell (**default-shell**) verändern. Einzige Voraussetzung ist, dass die Shell, die bei dem Kommando mit angegeben wird, auch in der Datei **/etc/shells** vorkommen muss. Man kann also nur auf jene Shells wechseln, die auch in der Datei **/etc/shells** aufgelistet sind.

```
chsh -s /Pfadzur/Shell Benutzer
```

Das ist die allgemeine Struktur des Kommandos.

```
[root@localhost root]#chsh -s /bin/tcsh helmut
```

Damit wird dauerhaft die Standard-Shell des Users helmut auf die tcsh eingestellt. Um wieder auf die ursprüngliche Standard-Shell – also die bash – zurückzuwechseln, geben wir das Kommando `chsh -s /bin/bash helmut` ein.

Um festzustellen, welche Kommandos man bisher eingegeben hat, kann man sich mit dem Befehl **history** die Datei **.bash\_history** formatiert ausgeben lassen. In dieser Datei werden alle Befehle, die man eintippt, gespeichert – nach dem **FIFO**-Prinzip (First In First Out) fallen die Befehle wieder heraus, wenn die Anzahl der zu speichernden Befehle oder die Dateigröße überschritten wurde. Diese Größen sind in Umgebungsvariablen definiert. Die Variablen sind **HISTSIZE** für die Anzahl der zu speichernden Befehle und **HISTFILESIZE** für die maximale Zeilenlänge der Datei **.bash\_history**.

```
echo $HISTFILESIZE
1000
echo $HISTSIZE
1000
```

Nachdem wir im vorangegangenen Kapitel einen ersten Einblick in ein Linux-System getan haben, wollen wir uns in den folgenden Abschnitten mit der Materie genauer auseinandersetzen. Um bei unserem bildlichen Vergleich zu bleiben - wir wollen ja nicht nur ein guter Fahrer des Systems werden, sondern wir wollen die Mechaniker des Systems sein. Los, riskieren wir einen ersten Blick unter die Motorhaube unseres Systems!

### 3.1

#### Das Dateisystem

Am Anfang stand das Dateisystem. Linux ist ein sehr offenes System und kennt ca. 90 unterschiedliche Dateisysteme. Was ist aber ein Dateisystem? Ein Dateisystem wird von jedem Betriebssystem benötigt, um Daten (blockweise) auf die Festplatte zu schreiben bzw. wieder auszulesen. Das heißt, eine Festplatte wird als blockorientiertes Gerät betrachtet. Die Form bzw. die Art und Weise, wie auf diesem blockorientiertem Gerät Dateien und Directories abgespeichert werden können, nennt man das Dateisystem, welches von Betriebssystem zu Betriebssystem unterschiedlich sein kann und ist. Bei Windows-Systemen ist derzeit das NTFS-Filesystem Standard, und im Linux-Bereich vollzieht sich gerade ein Paradigmenwechsel, der zweigeteilt ist. Im angloamerikanischen Raum geht der Trend hin zum **ext3**-Filesystem (extended Filesystem Version 3), und im europäischen Raum wird das **Reiser**-Filesystem (RFS) propagiert. Nichts desto trotz findet man in der „freien Wildbahn“ hauptsächlich den Vorgänger des ext3, das **ext2**-Filesystem. Das ext steht für extended und die Zahl für die Versionsnummer.

Wenn man sich nun eine Festplatte als Menge von Blöcken vorstellt, kann man diese – wie wir aus der Schule wissen – in Untermengen aufteilen. Das Aufteilen einer Festplatte in Untermengen nennt man Partitionieren, und die Untermengen selbst heißen Partitionen einer Festplatte. Dateisysteme werden dann nicht mehr über die gesamte Festplatte erstellt, sondern in den jeweiligen Partitionen. Beim Erstellen der Filesysteme (unter Windows nennt man das Formatieren) sollte schon berücksichtigt werden, welche Daten auf der Festplatte bzw. auf diesem Teil der Fest-



platte gespeichert werden sollen. Denn beim Erstellen des Filesystems wird auch die „physikalische“ Blockgröße des Dateisystems und damit die des Betriebssystems festgelegt.

**Tabelle 7: Unterstützte Dateisysteme**

<b><i>Dateisystem</i></b>	<b><i>Beschreibung</i></b>
ext	Vorgänger des ext2.
ext2	Standard Linux.
ext3	Erweiterung des ext2 um journaling Funktionalität.
iso9660	CD-ROM.
hpfs	OS/2.
msdos	MS-DOS.
nfs	Network File System (Unix-File-Sharing-System).
ntfs	Windows NT Filesystem.
smb	Server Message Block.
swap	Swap-Partition oder Datei (Auslagerungsdatei/Partition von Linux).
proc	Prozessverwaltung (wird vom System selbst erstellt und verwaltet).
vfat	Windows 95.
xfs	Das Dateisystem von Silicon Graphics.
u.v.m.	

Da blockweise auf die Festplatte geschrieben und gelesen wird, sollte der physikalische Datenblock der Festplatte, also die angegebene Blockgröße bei der Erstellung des Dateisystems, mit der Größe der Dateien, die darauf gespeichert werden in etwa übereinstimmen.

*Beispiel*

Nehmen wir an, wir wollen auf einer Partition nur kleine Bilder abspeichern, die alle eine Größe von ca. 4 kB haben. Wenn wir ein Dateisystem erstellen würden, welches einen physikalischen Datenblock von 1 kB hat, muss das System 4 Mal lesend/schreibend auf die Festplatte zugreifen, um nur ein Bild einlesen oder wegschreiben zu können. Ich verrate sicherlich kein Geheimnis, wenn ich sage, dass eines der langsamsten Geräte in einem Computer die Festplatte ist. Um eine deutliche Performancesteigerung zu erzielen, sollte das Dateisystem so angelegt werden, dass der physikalische Datenblock auch 4 kB beträgt, so kann ein einziger Lese/Schreib-Vorgang alles erledigen.

Der umgekehrte Fall ist auch nicht besser. Wenn wir einen physikalischen Datenblock von 4 kB haben und auf diesem Dateisystem nur Daten speichern, die eine Größe von 1 kB haben, dann benötigen wir zwar auch nur einen Lese/Schreib-Vorgang, allerdings vergeuden wir pro Datei 3 kB an Festplattenplatz. Ich kann somit nur 25% der möglichen Kapazität ausnützen.

Wenn noch Datenbanken mit im Spiel sind, wird es um so wichtiger, vorher einige Überlegungen anzustellen, anstatt wild darauf los zu werken.

**3.1.1****Das ext2 Filesystem**

Das **ext2fs** ist wohl derzeit das am weitesten verbreitete Dateisystem, welches man in Linux-Systemen findet. Die wichtigsten Eigenschaften des ext2-Dateisystems sind:

Dateinamen können bis zu 255 Zeichen lang sein. Dieses Limit kann aber bei Bedarf auf 1012 Zeichen erweitert werden. Die maximale Größe einer Datei ist auf 2GB beschränkt, kann aber auf 4 TB erweitert werden. Für den SuperUser wird standardmäßig 5% der Partitionsgröße reserviert, damit der Systemadministrator noch eingreifen kann, wenn ein Benutzer aus Versehen den gesamten Platz auf der Platte bzw. Partition verbraucht hat. Das Dateisystem arbeitet gebuffert. Das heißt, dass alle Schreib- und Lesevorgänge gebuffert verarbeitet werden, deshalb ist ein einmaliger Plattenscan (wie beim Suchen) etwas langsamer, aber weitere Scans (Suchvorgänge) sind wesentlich schneller, da der Inhalt bereits im Buffer steht. Das Schreiben auf die Festplatte arbeitet auch verzögert, also über Buffer. Gerade darin begründet sich ein wesentliches Merkmal, um Performancegewinne zu erzielen. Man überlässt dem System selbst, wann die gebufferten

Daten zu schreiben sind – am Bestem, wenn es gerade nicht viel zu tun hat. Die systemimmanenten Daten werden allerdings sofort geschrieben. Das gebufferte Schreiben hat den Nachteil, dass Daten bei einem Stromausfall korrupt werden können. Deshalb sollte man zu jedem Linux-Rechner eine unterbrechungsfreie Stromversorgung (USV) dazu stellen. Dieses Manko wird durch ein sogenanntes **journaling-Dateisystem** behoben. Was journaling bedeutet, wird etwas später beschrieben.

Das ext3-Dateisystem besitzt die gleichen Features wie das ext2 nur, dass dabei ein sogenanntes journaling eingeschaltet werden kann. Somit ist ein Wechsel von ext2 auf ext3 und umgekehrt jederzeit möglich.

Die Grundlage für das ext2-Dateisystem bilden die Komponenten: **Superblöcke** und **I-Nodes**.

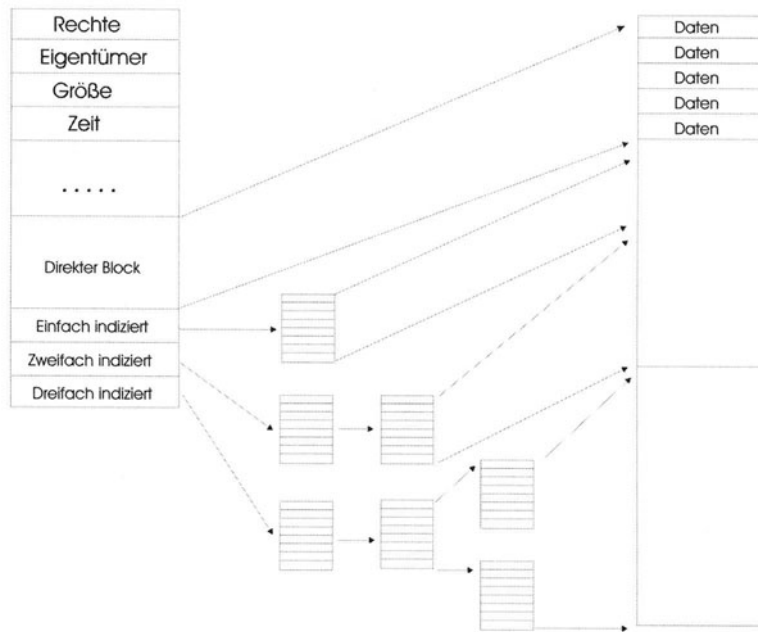
In den Superblöcken sind folgende Informationen abgespeichert:

- Anzahl der Datenblöcke.
- Größe der Blöcke.
- Name des Datenträgers (/dev/hda2).
- Maximale Anzahl von Dateien.
- Die Anzahl von I-Nodes und Zeigern auf den nächsten freien I-Node.
- Flags für die Dateistruktur.

In den I-Nodes sind folgende Werte abgelegt.

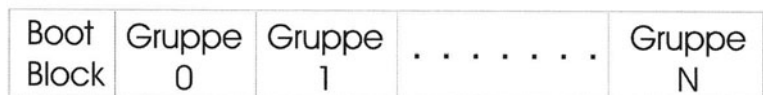
- Dateityp und Zugriffsrechte.
- Datei/Verzeichnisgröße.
- Erstellungsdatum/Modifikationsdatum/letzter Zugriff.
- Benutzer- und Gruppen-ID.
- Verweise auf zugehörige Datenblöcke.

Den Kernbestandteil bildet die I-Node-Tabelle. In dieser Tabelle sind die Verweise bzw. die Zeiger auf die eigentlichen Datenblöcke auf der Festplatte abgelegt. Den prinzipiellen Aufbau eines I-Nodes zeigt Abbildung 3.

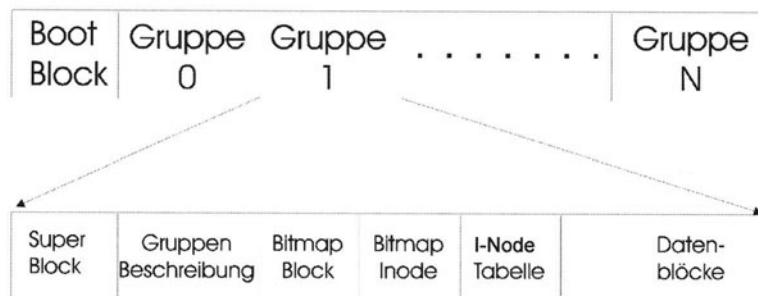


**Abbildung 3: Zusammenhang I-Node und Dateigröße**

Die Festplatte ist nun so aufgebaut:



**Abbildung 4: Aufbau der Festplatte**



**Abbildung 5: Einteilung der Gruppen**

Da die Superblocks und die I-Nodes die zentralen Elemente des Dateisystems darstellen, sind sie in Gruppen eingeteilt und über die Festplatte verteilt. Die Superblocks sind exakte Kopien. Da in ihnen Informationen über die Geometrie der Festplatte und alle wichtigen Informationen über das Dateisystem abgelegt sind, stellen sie die Grundlage dar, um überhaupt auf die Platte zugreifen zu können. Aus diesem Grund sind sie in Gruppen als redundante oder alternativ verwendbare Kopien gesichert.

Die Gruppendeskriptoren enthalten Informationen über die anderen Gruppen, damit ein Zusammenhalt und eine Wiederherstellung gewährleistet werden kann.

Die Bitmaps dienen nur dazu, freie Datenblöcke und I-Nodes auffinden zu können.

Bis zu dreifach indirekte Blöcke sind realisierbar, somit ist der theoretische Wert für die Dateigröße 8 TB bei einer Blockgröße von 8 kB ( $(12+1024+1024*1024+1024*1024*1024)*8\text{kb}$ ).

### 3.1.2

#### **ext3 – oder wie wird ein Journal geführt**

Wir wissen bereits, dass Linux mit der Architektur des verzögerten Schreibens einiges an Performance gewinnt. Allerdings haben wir auch erkannt, dass bei einem Stromausfall (oder durch unbedachtes Ausschalten) des Computers die Daten korrupt werden können. Da dies systembedingt ist, gibt es Tools und Programme, die man in einem solchen Fall einsetzen kann, um das Dateisystem wieder zu reparieren. Bei jedem Start des Systems wird automatisch festgestellt, ob eine Reparatur notwendig ist oder nicht. Man braucht kein Prophet zu sein, um festzustellen, dass es bei großen Festplattenkapazitäten und **RAID**-Systemen mitunter sehr, sehr lange dauern kann, bis das Dateisystem wieder hergestellt ist.

Als Abhilfe haben wir bereits das sogenannte journaling-Dateisystem beschworen.

In einer Art Datenbank – dem Journal – werden alle offenen Dateien protokolliert. Sobald die Datei wieder geschlossen wird, wird sie auch wieder aus dem Journal gestrichen. Wenn nun ein Computer mit aktivem Journaling von einem Crash betroffen ist, muss man nicht mehr warten, bis die gesamte Platte durchforstet wurde. Denn in diesem Fall hat sich das Journal gemerkt, welche

Dateien zu diesem Zeitpunkt geöffnet<sup>1</sup> waren, und es brauchen nur noch diejenigen Bereiche und Partitionen bearbeitet zu werden, die eine Entsprechung im Journal haben. Dass dies um vieles schneller von statten geht, ist leicht einsichtig.

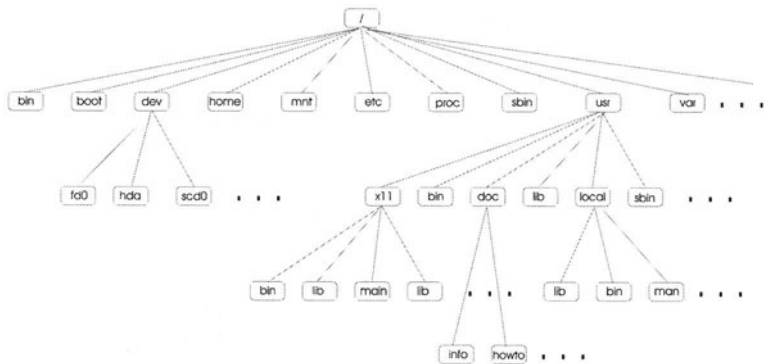
Das ext3-Dateisystem ist ein Vertreter eines journaling-Dateisystems. Ein weiterer wichtiger Vertreter von journaling-Dateisystemen ist das Reiser-Filesystem (Reiserfs).

Allerdings ist das ext3 mit dem ext2-Dateisystem vollständig kompatibel bis auf das Journaling. Dieses kann man aber beim ext3 ein- und ausschalten. So kann man bequem von einem auf das andere umgeschaltet werden.

## 3.2

### Die Systemarchitektur

Bevor wir uns die Festplatte untertan machen können, sollten wir verstehen, wie Linux mit Dateien, Verzeichnissen und diversen Hardware- und Peripheriegeräten umgeht. Dazu springen wir ins kalte Wasser und betrachten einen typischen Ausschnitt aus einer Verzeichnisstruktur.



**Abbildung 6: Ausschnitt aus einem Verzeichnisbaum**

Wir erkennen auf den ersten Blick, dass diese Verzeichnisstruktur wie ein auf den Kopf gestellter Baum aussieht. Diese Architektur nennt man deshalb auch Baumstruktur oder hierarchische Verzeichnisstruktur, die mit einer **Wurzel (ROOT)** beginnt und

<sup>1</sup> Natürlich können nur geöffnete Dateien von einer möglichen Inkonsistenz betroffen sein, da geschlossene (oder read only) Dateien nicht modifiziert werden können.

dann wie Äste auseinander wächst. Die Äste entsprechen den Directories oder Verzeichnissen, und die Blätter werden durch die Dateien repräsentiert. Als Zeichen für den Anfang des Verzeichnisbaums, also für die Wurzel, wird der Slash / verwendet. Was uns dabei noch aufgefallen ist, es gibt weit und breit in dem Verzeichnisbaum keinen Laufwerksbuchstaben, so wie unter Windows gewohnt. In Windows entspricht jede Partition einer jeden Festplatte einem anderen Laufwerksbuchstaben. Bei Linux hingegen sind die Partitionen für den User völlig transparent.

Das heißt, die Festplatten und deren Partitionen werden irgendwo in das Verzeichnissystem „eingehängt“. Auf den Bereich der Partition wird zugegriffen, wenn man in das Verzeichnis, auf das die Partition eingehängt wurde, hinein wechselt. Das bedeutet aber, dass auf alle Fälle in die Wurzel eine Partition gehängt werden muss, denn sonst würde der Baum gar nicht erst entstehen können. Die weiteren Partitionen werden der Wurzel folgend als Äste in den Baum eingehängt.

Wie greift das Betriebssystem auf die Partitionen zu, wenn es keine Laufwerksbuchstaben mehr gibt?

Unter Linux wird jegliche Hardware durch ein sogenanntes **Pseudodevice** repräsentiert. Dieses Pseudodevice wird durch eine spezielle Datei repräsentiert. Das System spricht die gesamte Hardware und damit auch die Partitionen einer Festplatte durch diese speziellen **Geräte-Dateien** an. Damit haben wir auch gleich einen wesentlichen Unterschied zu Windows-Systemen gefunden. Die Unterstützung der Hardware steckt vollkommen im Kernel oder in Kernel-Modulen. Das heißt, im Normalfall genügt es, die Hardware an den Computer anzuschließen, und sie wird unterstützt und funktioniert. Natürlich muss sie noch konfiguriert werden. Das bedeutet, dass bei den gängigen Distributionen eine sehr große Anzahl an unterstützter Hardware und Peripheriegeräte bereits bei der Erstellung des Kernels – entweder statisch oder als Modul – berücksichtigt wurden.

Es ist im allgemeinen nicht notwendig, einen „Treiber“ zu installieren. Wenn man allerdings einmal an eine Hardware gelangt, die von meiner aktuellen Distribution nicht unterstützt wird, ist es notwendig den Kernel neu zu kompilieren oder ein neues Kernel-Modul einzubinden – siehe dazu Kapitel 7.

Welche speziellen Geräte-Dateien sind für den Zugriff auf die Festplatten zuständig, und wie werden die Partitionen unterschieden?

Alle Geräte-Dateien haben einen bestimmten Platz in der Verzeichnishierarchie. Sie alle sind im Zweig **/dev** gespeichert. Bei unserer derzeit benutzten Installation sind dies z. B.: 7520 verschiedene Geräte-Dateien.

Die Festplatten (IDE) werden mit **hd** für Hard-Disk und mit einem Buchstaben a, b, c, ... bezeichnet, je nachdem, ob es sich um die erste oder zweite oder dritte ... Festplatte handelt.

#### Einschub

In einem herkömmlichen PC gibt es zwei IDE-Bus-Controller, die jeweils zwei Geräte (Festplatten, CD-ROM, DVD, ...) bedienen können. Zur Unterscheidung der Geräte auf einem **IDE-Bus** werden sie entweder als **Master** oder als **Slave** gejumpert. Damit ist das erste Gerät jenes, das auf dem ersten IDE-Controller angeschlossen ist und als Master gejumpert wurde.

Das heißt, eine Festplatte, die auf dem ersten IDE-Controller angeschlossen ist und als Master gejumpert wurde, hat im Linux-System die Entsprechung **/dev/hda**. Eine Festplatte, die auf dem zweiten IDE-Controller angeschlossen ist und als Slave gejumpert wurde, heißt unter Linux dann **/dev/hdb**.

Damit können wir schon die einzelnen Festplatten ansprechen. Nun wollen wir die einzelnen Partitionen ansprechen können, und das funktioniert über eine fortlaufende Nummer, so wie die Partitionen angelegt wurden. Also für die erste Partition auf der Festplatte die **1** und für die zweite Partition die **2** u.s.w.

Damit ist der vollständige Name der vierten Partition einer Festplatte, die am zweiten IDE-Controller angeschlossen ist und als Master gejumpert wurde, **/dev/hdb4**.

Hier ein kleiner Auszug der wichtigsten Geräte-Dateinamen:

**Tabelle 8: Gerätenamen**

<b>Name</b>	<b>Bedeutung</b>
hd	IDE-Harddisk.
sd	SCSI-Disk.
fd	Floppy Disk.
st	SCSI Tape Drive.
tty	Terminal.



lp	Drucker.
pty	Remote Terminal.
midi	Midi Ports.
ttyS	Serieller Anschluss.
cua	Entsprechen COM Ports.

Bei allen werden die einzelnen Geräte durch Nummern und / oder Buchstaben, die an die Geräte-Dateien angehängt werden, unterschieden.

- /dev/hda2 Zweite Partition der ersten Festplatte.
- /dev/sdb1 Erste Partition der zweiten SCSI-Festplatte.
- /dev/fd0 Erste Floppy Disk.
- /dev/ttyS0 Entspricht dem ersten Seriellen Anschluss.
- /dev/cdrom Ist das erste CD-ROM-Laufwerk.

Da CD-ROM-Laufwerke wie Festplatten, als blockorientierte Geräte behandelt werden, kann man auch auf die CD-ROM, wenn es am zweiten IDE-Controller als Master konfiguriert wurde z. B.: mittels /dev/hdc zugreifen. Damit haben wir das nötige Fachwissen, um eine Partition und darauf ein Dateisystem zu erstellen.

### 3.2.1

#### Partitionieren und Ordnung schaffen ...

... oder wie zerstückelt man eine Festplatte und ordnet sie wieder.

Um eine Festplatte zu partitionieren, verwenden wir das Programm **fdisk**. Wir geben auf der Kommandozeile – ich werde ab nun anstelle des gesamten Prompts immer das Zeichen # als Promptkurzform verwenden – folgendes ein:

```
#fdisk /dev/hdb
```

Die Anzahl der Zylinder für diese Platte ist auf 4865 gesetzt.

Daran ist nichts verkehrt, aber das ist größer als 1024 und kann in bestimmten Konfigurationen Probleme hervorrufen mit:

- 1) Software, die zum Bootzeitpunkt läuft (z. B. ältere LILO-Versionen)
- 2) Boot- und Partitionierungssoftware anderer Betriebssysteme  
(z. B. DOS FDISK, OS/2 FDISK)

Befehl (m für Hilfe):

Wenn wir der Aufforderung nachkommen und ein m eingeben, erhalten wir folgenden Bildschirm-

Befehl	Bedeutung
a	(De)Aktivieren des bootfähig-Flags
b	»bsd disklabel« bearbeiten
c	(De)Aktivieren des DOS Kompatibilitätsflags
d	Eine Partition löschen
l	Die bekannten Dateisystemtypen anzeigen
m	Dieses Menü anzeigen
n	Eine neue Partition anlegen
o	Eine neue leere DOS Partitionstabelle anlegen
p	Die Partitionstabelle anzeigen
q	Ende ohne Speichern der Änderungen
s	Einen neuen leeren »Sun disklabel« anlegen
t	Den Dateisystemtyp einer Partition ändern
u	Die Einheit für die Anzeige/Eingabe Ändern
v	Die Partitionstabelle überprüfen
w	Die Tabelle auf die Festplatte schreiben und das Programm beenden
x	Zusätzliche Funktionen (nur für Experten)

Befehl (m für Hilfe):

Wir wollen eine neue Partition auf der zweiten Festplatte hdb anlegen.

```
#fdisk /dev/hdb
```

```
Befehl (m für Hilfe): n
```

```
Befehl  Aktion
```

```
  e      Erweiterte
```

```
  p      Primäre Partition (1-4)
```

```
p
```

```
Partitionsnummer (1-4): 1
```

```
Erster Zylinder (1-4998, Vorgabe: 1): 1
```

```
Letzter Zylinder oder +Größe, +GrößeK oder +GrößeM (1-4998, Vorgabe: 4998): +2G
```

```
Befehl (m für Hilfe): p
```

```
Platte /dev/hdb: 41.1 GByte, 41110142976 Byte
```

```
255 Köpfe, 63 Sektoren/Spuren, 4998 Zylinder
```

```
Einheiten = Zylinder von 16065 * 512 = 8225280 Bytes
```

Gerät	boot.	Anfang	Ende	Blöcke	Id	Typ
/dev/hdb1		1	244	1959898+	83	Linux

```
Befehl (m für Hilfe): w
```

Die Partitionstabelle wurde verändert!

Calling ioctl() to re-read partition table.

Syncing disks.

Damit haben wir eine Partition in der Größe von 2 GB auf der zweiten IDE-Platte in unserem System erzeugt. Noch ist diese allerdings nicht einsatzbereit. Wir müssen# noch ein Dateisystem auf der Partition erstellen. Unter Windows würde man sagen: „Die Partition gehört noch formatiert“.

Zum Erstellen eines Dateisystems dient der allgemeine Befehl **mkfs** (make filesystem). Für die gängigsten Dateisysteme wie ext2, ext3, Reiserfs, swap, ... existieren spezielle Tools wie: **mke2fs**, **mkreiserfs**, **mkswap**, **mkfs.ext3**, **mkfs.ext2**.

Wir wollen nun ein ext3-Dateisystem auf unserer Partition erstellen und verwenden dazu **mkfs.ext3**. Im einfachsten Fall können wir also schreiben:

```
[root@localhost root]# mkfs.ext3 /dev/hdb1
```

```
mke2fs 1.32 (09-Nov-2002)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
245280 inodes, 489974 blocks
24498 blocks (5.00%) reserved for the super user
First data block=0
15 block groups
32768 blocks per group, 32768 fragments per group
16352 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912
```

```
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

This filesystem will be automatically checked every 21 mounts or 180 days, whichever comes first. Use **tune2fs -c** or **-i** to override.

Voilà - damit haben wir ein ext3 mit Defaultwerten erstellt. Das heißt, wir haben eine Blockgröße von 4kB, 5% Reserve für den SuperUser, ein Journal, und unser Superblock wiederholt sich 5 mal.

**Tabelle 9: Optionen für mkfs**

<b>Option</b>	<b>Bedeutung</b>
-c	Überprüft das Gerät auf defekte Blöcke.
-b	Blockgröße.
-i	Bytes per I-Node (sollte nie kleiner sein als die Blockgröße).
-j	Erzeugt ein Journal.
-J	Journal-Optionen.
-L	Platten-Lable.
-m	Reservierter Platz in %.
-n	Erzeugt kein Dateisystem, sondern zeigt nur an, was es tun würde.
-R	Raid-Optionen.

Wenn wir eine Partition mit einem Dateisystem haben, gehört dieses natürlich von Zeit zu Zeit überprüft, ob die Daten, die darauf gespeichert sind, auch wirklich noch konsistent sind. Dafür gibt es wieder spezialisierte Tools wie **e2fsck** bzw **fsck.ext2**, **fsck.ext3**, **fsck.reiser**, **fsck.vfat**, ...

Für das Tool fsck.ext3 sind die wichtigsten Optionen die folgenden:

**Tabelle 10: Optionen für fsck**

<b>Option</b>	<b>Bedeutung</b>
-b	Benutzt zur Überprüfung einen anderen Superblock als den ersten. Nützlich wenn der erste Superblock zerstört wurde.
-c	Testet auf defekte Blöcke.
-D	Optimiert die Verzeichnisstruktur durch eventuelles Neuindizieren.
-f	Erzwingt den Check, auch wenn alles o.k. erscheint.

-l	Angabe einer Datei mit Bad Blocks.
-n	Fügt automatisch ein n0 auf alle Fragen ein.
-y	Damit werden alle Fragen mit yes beantwortet.
-p	Repariert das Dateisystem ohne Fragen.
-v	Verbose-Modus.

**Wichtig**

Bevor man einen Dateisystem-Check durchführen kann, muss die Partition bzw. das Dateisystem offline sein. Wir werden dazu später unmounted sagen.

Der Dateisystem-Check läuft in 5 Durchgängen ab und zwar:

1. Durchgang: Überprüft die I-Nodes, Blöcke und Größe.
2. Durchgang: Überprüft die Verzeichnisstruktur.
3. Durchgang: Überprüft den Verzeichniszusammenhang.
4. Durchgang: Überprüft den Referenzzähler.
5. Durchgang: Überprüfung der Gruppensummen Informationen.

**Tabelle 11: Rückgabewerte von fsck**

<b>Wert</b>	<b>Bedeutung</b>
0	Kein Fehler.
1	Dateisystemfehler wurden korrigiert.
2	Wie 1, jedoch ist ein Neustart empfehlenswert.
4	Dateisystemfehler blieben unkorrigiert.
8	Operational Error.
16	Syntax-Fehler.
32	Überprüfung wurde vom User unterbrochen.
128	Fehler in den Shared Libraries.

Wenden wir dieses Wissen auf unsere neu erstellte Partition an. Wir erwarten allerdings, dass keine Fehler vorhanden sind, und verwenden deshalb die Option `-f`. Als Ergebnis erhalten wir:

```
[root@localhost root]# fsck.ext3 -f /dev/hdb1

e2fsck 1.32 (09-Nov-2002)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/test: 41/26104 files (2.4% non-contiguous),
12577/104391 blocks
```

Mit dem **debugfs**-Tool kann man interaktiv das Filesystem debuggen.

**Tabelle 12: Optionen für debugfs**

<b>Option</b>	<b>Bedeutung</b>
<code>-w</code>	Öffnet das Dateisystem im Schreib-Lese-Modus.
<code>-c</code>	Hier wird das Dateisystem in einem catastrophic Modus geöffnet, was bei schwer beschädigten Dateisystemen hilfreich sein kann.
<code>-s BLOCK</code>	Erzwingt das Lesen des Superblocks von der angegebenen Stelle.
<code>-b BLOCK</code>	Erzwingt das Öffnen mit der angegebenen Blockgröße.

Mit dem Befehl **tune2fs** kann man die Parameter eines ext2-Dateisystemes verändern und auch zwischen einem ext2 und einem ext3 hin und her wechseln.

**Tabelle 13: Optionen für tune2fs**

<b>Option</b>	<b>Bedeutung</b>
-c MAX	Verändert das Maximum von Mountvorgängen, bevor ein Filesystemcheck notwendig wird.
-f	Zwingt das tune2fs-Kommando dazu, seine Arbeit abzuschließen, auch wenn Fehler auftreten.
-i Intervall	Gibt das Intervall zwischen zwei Filesystemchecks an.
-j	Fügt ein ext3-Journal hinzu.
-J Optionen	Ändert die Standardjournaloptionen. Optionen können sein: Size-Größe des Journals. Device=externes device.
-l	Listet den Inhalt des Filesystemsuperblocks auf.
-L Label	Dateisystemlabel.
-m Proz	Prozentuale Angabe Proz des reservierten Bereiches.
-T	Zeit, an der zuletzt ein Check durchgeführt wurde.
-u	Benutzer, der den reservierten Bereich nutzen darf.

#tune2fs -c 200 /dev/hda5, stellt die maximale Anzahl von mounts zwischen zwei Dateisystemüberprüfungen auf 200 ein.

Nun wollen wir das neue Dateisystem auch nutzen können, denn es ist noch immer offline, also unmounted. Wir werden uns im nächsten Abschnitt die notwendigen Programme und Tools erarbeiten, die wir dazu benötigen.



### 3.2.2 Die wichtigsten Befehle

Bei der Arbeit mit jedem Betriebssystem werden sich sehr bald die wichtigsten Befehle derart darstellen, dass man sie am häufigsten benutzt. Ganz ohne Zweifel sind dies unter Linux die Befehle **ls**, **pwd**, **cd**, **mkdir** und **touch**.

Der Befehl **ls** (list) listet den Inhalt des aktuellen Verzeichnisses, in dem ich mich gerade befinde, auf.

```
[root@localhost root]# ls
anaconda-ks.cfg buch fdisk.txt fsck.txt install.log in-
stall.log.syslog
```

Wenn wir die Option **-l** für Long-Format dazu angeben, erhalten wir folgende Ausgabe:

```
[root@localhost root]# ls -l
insgesamt 48
-rw-r--r-- 1 root root 1310 23. Jul 14:41 anaconda-ks.cfg
drwxr-xr-x 2 root root  4096 24. Jul 09:37 buch
-rw-r--r-- 1 root  root  2977 28. Jul 14:14 fdisk.txt
-rw-r--r-- 1 root  root   335 29. Jul 10:27 fsck.txt
-rw-r--r-- 1 root  root 24815 23. Jul 14:27 install.log
-rw-r--r-- 1 root  root  3082 23. Jul 14:27 install.log.syslog
```

Wenn wir die Option **-a** dazu angeben, werden die versteckten Dateien auch mit angezeigt.

*Einschub*

Unter Linux werden Dateien, die mit einem Punkt beginnen, als versteckte Dateien behandelt. Das bedeutet, dass sie mit dem normalen **ls** Befehl ohne Angabe der Option **-a** nicht angezeigt werden. Die versteckten Dateien sind meistens Systemdateien wie Konfigurationsdateien. Wir haben solche Dateien bereits kennengelernt<sup>2</sup>.

```
[root@localhost root]# ls -la
insgesamt 232
drwxr-x--- 15 root  root  4096 29. Jul 10:48 .
drwxr-xr-x 19 root  root  4096 29. Jul 09:31 ..
-rw-r--r--  1 root  root  1310 23. Jul 14:41 anaconda-ks.cfg
```

---

<sup>2</sup> Vergleiche Seite 27

```

-rw----- 1 root    root    4239 28. Jul 14:31 .bash_history
-rw-r--r-- 1 root    root      24 10. Jun 2000 .bash_logout
-rw-r--r-- 1 root    root    261 24. Jul 14:55 .bash_profile
-rw-r--r-- 1 root    root    176 23. Aug 1995 .bashrc

```

**Tabelle 14: Optionen für ls**

<b>Option</b>	<b>Bedeutung</b>
-a	Alles.
-B	Einträge, die mit ~ enden, werden nicht ausgegeben.
-G	Gruppenangaben unterdrücken.
-h	Human Readable.
-i	Ausgabe der I-Nodes.
-k	Ausgabe in kB-Blöcken.
-l	Langes Listenformat.
-m	Ausgabe mit Kommata getrennt.
-n	Gruppen und User in numerischer Schreibweise.
-O	Langes Listenformat ohne Gruppeninformationen.
-Q	Ausgaben werden unter doppelten Hochkommata ausgegeben ("").
-R	Rekursive Ausgabe von Verzeichnissen.
-S	Nach Dateigröße sortieren.
-r	Umkehren der Sortierung.
-t	Sortiert nach Änderungszeit.
-u	Sortiert nach Zugriffszeit.
-l	Listet eine Datei pro Zeile auf.
--help	Hilfetext.

Wenn wir in einem Verzeichnis alle Dateien ausgeben wollen, die die Endung .jpg haben, kann ich den Wildcardcharakter \* verwenden:

```
[root@localhost root]# ls *.jpg
hans.jpg  sepp.jpg  susi.jpg
```

### *Einschub*

Wir haben schon einige Befehle auf der Kommandozeile eingegeben. Wir haben uns aber über eine eventuelle Korrektheit der Eingabe noch keine Gedanken gemacht. So wie alles in der Computerei, unterliegt auch die Art und Weise, wie man auf der Kommandozeile einen Befehl eingibt, gewissen Regeln. Eine korrekte Kommandozeileneingabe hat folgendes Format:

#### **Befehl      Optionen      Argumente**

Die Reihenfolge ist dabei wichtig. Dass man immer alle drei Elemente mit angibt, ist nicht notwendig. Betrachten wir dazu als Beispiel den ls-Befehl:

- Nur als Befehl.  
ls
- Befehl mit der Option -la.  
ls -la
- Vollständig, also auch noch mit dem Argument \*.jpg.  
ls -la \*.jpg

Als Options- und **Argument-Trenner** dient dabei der Leerschritt oder das **Space**. Deshalb sollten Sie keine Leerzeichen in Dateinamen verwenden, denn wenn Sie z. B. den Dateinamen Buchhaltung Seppi haben und ein ls -l darauf anwenden wollen, also folgende Kommandozeile eingeben wollen:

```
#ls -ls Buchhaltung Seppi
```

bekommen wir eine Fehlermeldung:

```
ls: Seppi: Datei oder Verzeichnis nicht gefunden
```

Dies kommt daher, dass Linux das Leerzeichen als Optionen- und Argument-Trenner verwendet und damit zwei Argumente – also ein Argument Buchhaltung und ein Argument Seppi – erkennt, und wir haben natürlich keine Datei, die Seppi heißt.

Sie wissen bereits, wie man dies dennoch erfolgreich tun kann. Ja, mit dem \ Metacharakter. Dieser hebt die Bedeutung des nachfolgenden Zeichens auf, und deshalb können wir schreiben:

```
#ls -ls Buchhaltung\ Seppi
```

Damit wird aus dem Optionen- und Argument-Trenner ein einfaches Leerzeichen, welches zum Dateinamen gehörig interpretiert wird.

### *Einschub*

Wenn man die Optionen des `ls`-Befehls und die erhaltenen Ergebnisse genauer betrachtet, fallen einige Ungereimtheiten auf. Diese resultieren aus dem **Alias-Mechanismus**, der Shell. Mit dem Alias-Mechanismus kann man eigene Befehlskürzel für längere Befehle samt Optionen setzen. Um festzustellen, welche **Aliases** schon gesetzt wurden, gibt man einfach den Befehl **alias** ein.

```
# alias
alias cp='cp -i'
alias l.='ls -d .* --color=tty'
alias ll='ls -l --color=tty'
alias ls='ls --color=tty'
alias mv='mv -i'
alias rm='rm -i'
alias vi='vim'
```

Und da können wir erkennen, dass, unter anderem auch der Befehl `ls` ein Kürzel für `ls -color = tty` darstellt und damit die Farbeinstellung automatisch eingeschaltet ist, was eigentlich nicht Standard wäre. Es wird bei der Eingabe von `ls` nicht der Befehl selbst, sondern das Kürzel ausgeführt.

Wir können auch unsere eigenen Kürzel setzen. Lassen Sie uns Folgendes probieren:

```
#alias my='ls -la'
```

Wir erhalten durch einfache Eingabe von `my` dieselbe Ausgabe wie mit `ls -la`. Um einen Alias wieder auszuschalten, verwendet man den Befehl **unalias**. Als Option bzw. Parameter für diesen Befehl ist der Name des Aliases, den man löschen möchte. Wenn wir unseren Alias `my` wieder löschen wollen, dann setzen wir einfach Folgendes ab:

```
#unalias my
```

*Frage* In welchen Dateien werden die Aliases gesetzt werden?  
In der Datei `/etc/profile` für alle Benutzer des Systems und in `./bashrc` in dem jeweiligen persönlichen Verzeichnis des Benutzers.

*Hinweis* Die persönlichen Verzeichnisse – auch **HOME**-Verzeichnisse genannt – aller Benutzer sind defaultmäßig, wenn man beim Anlegen des Benutzers nicht etwas Anderes angibt, in dem Verzeichnis **/home** angesiedelt und heißen genauso wie der Benutzer – bzw. dessen Account. Das persönliche Heimatverzeichnis des Benutzers `helmut` heißt also **/home/helmut**. Nach dem Anmelden an das System ist das aktuelle Verzeichnis des Benutzers sein HOME-Verzeichnis. Nur der Benutzer selbst hat das Recht, in das Verzeichnis zu wechseln. Eine Ausnahme bildet der SuperUser `root`. Das persönliche Verzeichnis, das HOME-Verzeichnis des `root`-Users, ist nicht im `/home` angesiedelt, sondern in **/root**.

Der nächste wichtige Befehl für das Verzeichnissystem ist der Befehl ***pwd*** (present working directory). Er zeigt mir das aktuelle Verzeichnis, in dem ich mich gerade befinde an.

```
#pwd  
/root
```

Um sich in dem Verzeichnisbaum herumzubewegen, gibt es den Befehl ***cd*** (change directory).

Rufen wir uns die Verzeichnisstruktur wieder ins Gedächtnis. Wenn wir dieses genauer betrachten und von unserem Verzeichnis `/root` in das Verzeichnis `/usr/local/bin` wechseln wollen, dann erkennen wir sofort, dass wir zwei Möglichkeiten haben, den Weg dorthin zu beschreiben. Entweder wir bezeichnen den Weg ***absolut*** von der Wurzel `/` (ROOT) weg oder wir beschreiben den Weg ***relativ*** von unserem aktuellen Verzeichnis aus.

Um einen relativen Pfad zu erstellen, benötigt man Bezeichnungen für das aktuelle Verzeichnis und für das übergeordnete Verzeichnis.

- `./`      Aktuelles Verzeichnis.
- `../`     Übergeordnetes Verzeichnis.

Wenn man in der Verzeichnishierarchie aufsteigen möchte, also in das übergeordnete Verzeichnis wechseln will, schreibt man `../`, wenn man zwei Verzeichnisebenen hochsteigen möchte, schreibt man `../../` und so weiter. Wenn man vom aktuellen Verzeichnis z. B. in das Verzeichnis `doku` absteigen will, schreibt man `./doku`.

**Tabelle 15: Pfadvarianten**

<i><b>Pfadangabe</b></i>	<i><b>Bedeutung</b></i>
<code>/usr/local/bin</code>	Absoluter Pfad (beginnt <i><b>immer</b></i> mit einem <code>/</code> ).
<code>../usr/local/bin</code>	Relativer Pfad (Beginnt <i><b>immer</b></i> mit <code>./</code> oder <code>../</code> ).

Fassen wir noch einmal zusammen: Man kann in das Verzeichnis `/usr/local/bin` mit folgenden Befehlen gelangen:

```
#cd /usr/local/bin           -- Absoluter Pfad
oder
#cd ../usr/local/bin         -- Relativer Pfad
```

Damit wir eigene Verzeichnisse anlegen können, benötigen wir den Befehl ***mkdir*** (make directory).

**Tabelle 16: Optionen für mkdir**

<i><b>Optionen</b></i>	<i><b>Bedeutung</b></i>
<code>-m</code>	Setzt die Zugriffsrechte.
<code>-p</code>	Erzeugt auch die übergeordneten Verzeichnisse, wenn nicht vorhanden.
<code>--help</code>	Hilfe.

Auf die Zugriffsrechte gehen wir bei der Benutzerverwaltung<sup>3</sup> ein. Das Programm `mkdir` arbeitet in der Defaulteinstellung so, dass man nur ein Verzeichnis anlegen kann und zwar in dem Verzeichnis, in dem man sich gerade befindet.

```
#pwd
/root
#mkdir buchhaltung
#cd buchhaltung
#pwd
/root/buchhaltung
#mkdir lager
#cd lager
#pwd
/root/buchhaltung/lager
```

Dies können wir aber auch in einem einzigen Schritt erledigen und zwar mit der Kommandozeile:

```
#mkdir -p /root/buchhaltung/lager
```

### Frage

Versuchen Sie die Elemente Befehl - Optionen - Argumente zu beschreiben.

Um die Datei `september.txt` (eine leere Datei) in dem Verzeichnis `/root/buchhaltung/lager` zu erstellen, verwenden wir den Befehl ***touch***.

```
#touch september.txt
#ls
september.txt
```

Wenn man den Befehl `touch` ohne Optionen ausführt, legt man eine leere Datei an. Der Befehl `touch` kann allerdings noch mehr.

---

<sup>3</sup> Vergleiche Kapitel 4.

**Tabelle 17: Optionen für touch**

<b>Option</b>	<b>Bedeutung</b>
-a	Ändert die Zugriffszeit.
-C	Keine Datei erzeugen.
-m	Ändert die Modifikationszeit.
-r datei	Ändert die Zeit auf die der angegebenen Datei anstelle der aktuellen Systemzeit.
-t marke	Zeitangabe in der Form [[HH]JJ]MMTTSSmm[.ss] anstelle der aktuellen Systemzeit.
--help	Hilfe.

Um die leere Datei mit Leben zu erfüllen, kann man den auf jedem Linux-System vorhandenen Editor **vi** (visual editor) verwenden.

### 3.2.3

#### Der vi – alt aber gut

Der vi ist ein bildschirmorientierter Editor, der bereits seine 30 Jahre auf dem Buckel hat, aber dafür den Vorteil besitzt, dass er auf einem jeden Linux-System zu finden ist. Dies soll natürlich nicht seine Leistungsfähigkeit schmälern. Ich selbst bin Minimalist, und deshalb freut es mich persönlich, dass ich mir nur einen Editor und seine Bedienung merken muss.

Der Programmaufruf erfolgt mit `vi dateiname`.

z. B.:

```
#vi /root/buchhaltung/lager/myfile.txt
```

So kann ich eine Datei namens `myfile.txt` im Verzeichnis `/root/buchhaltung/lager` editieren oder auch neu anlegen lassen, falls sie noch nicht existiert.

Wenn man eine neue Datei namens `myfile.txt` im vi öffnet, erhält man folgenden Bildschirm.



~

~

~

~

"myfile.txt" [Neue Datei]

Die ~ (Tilde-Zeichen) zeigen das Dateiende an. In der untersten Zeile des Bildschirms wird der Dateiname angezeigt, und dass es sich dabei um eine neu angelegte Datei handelt.

Jetzt befinden wir uns in einem der drei Modi, die der vi kennt, und zwar im **Kommand**-Modus.

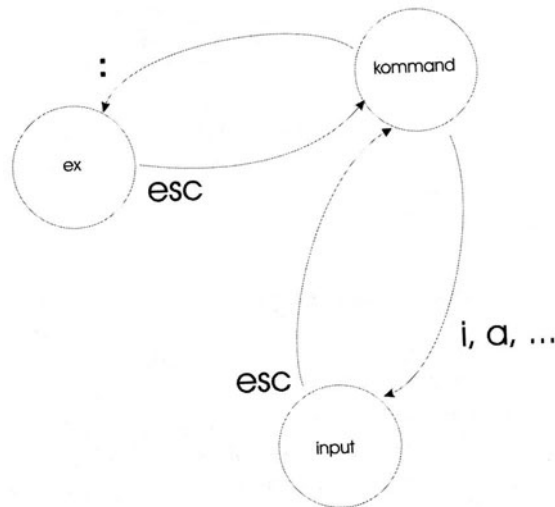


Abbildung 7: Die 3 Modi des vi

Im Kommand-Modus kann ich den Cursor in der Datei bis zum Dateiende frei bewegen und Befehle fürs Kopieren, Löschen u.s.w. ausführen. Im sogenannten **ex**-Modus kann ich die Datei z. B. speichern, eine neue Datei einlesen, suchen oder ersetzen. Zum eigentlichen Schreiben gibt es den **Insert**- oder **Input**-Modus. In diesem kann man nur schreiben. Um von all diesen Modi wieder zurückzukehren zum Ausgangsmodus, also dem Kommand-Modus, betätigt man die **ESC-Taste**.

Betrachten wir zuerst den ex-Modus. In diesen Modus gelangt man, wenn man den **:** drückt. Es erscheint in der untersten Zeile ein Doppelpunkt. Nun kann man den eigentlichen Befehl oder Auftrag eingeben.

**Tabelle 18: Ex-Befehle**

<b><i>Befehl</i></b>	<b><i>Bedeutung</i></b>
:w	Schreibt den Buffer in die Datei, also die Datei auf die Festplatte.
:w!	Hierbei wird ein eventueller Schreibschutz ignoriert.
:q	Beendet die Bearbeitung. Man muss zuvor gespeichert haben. Wenn man aussteigen will, ohne die Änderungen zu speichern, verwendet man wieder das „Fehler ignorieren“-Zeichen, den <b>!</b> also <b>:q!</b> .
:wq	Schreiben und den vi beenden.
:x	So wie :wq.
:w Name	Schreibt den Bufferinhalt in die Datei Name (entspricht „Datei speichern unter ...“).
:r Name	Liest die Datei Name an der aktuellen Cursor-Position ein.
:n	Wechselt zur nächsten geladenen Datei, wenn beim Aufruf mehrere Dateien angegeben wurden.

Aus dem ex-Modus kommt man durch Betätigen der ESC-Taste wieder zurück in den Kommand-Modus.

Um in den Eingabe-Modus zu wechseln, kann man folgende Tasten bzw. Kommandos verwenden:

**Tabelle 19: Kommand-Befehle**

<b><i>Befehl</i></b>	<b><i>Bedeutung</i></b>
a	Append - Eingabe rechts vom Cursor.
A	Append – Eingabe am Zeilenende.
i	Insert – Eingabe links vom Cursor.
I	Insert – Eingabe am Zeilenanfang.
o	Eingabe in der nächsten Zeile – fügt neue Zeile ein.
O	Eingabe in der vorherigen Zeile – fügt neue Zeile ein.

Aus dem Eingabe-Modus kann man mit Betätigen der Taste ESC wieder in den Kommand-Modus wechseln.

Im Kommand-Modus stehen uns eine Vielzahl an Befehlen zur Verfügung, die, wie Sie bald sehen werden, einem gewissen Schema folgen. Die meisten Befehle wirken immer ab der aktuellen Cursorposition.

**Tabelle 20: Befehle für die Cursorbewegung**

<b><i>Befehl</i></b>	<b><i>Bedeutung</i></b>
h	Ein Zeichen nach links.
l	Ein Zeichen nach rechts.
k	Ein Zeichen nach oben.
j	Ein Zeichen nach unten.
H	Anfang erste Zeile des Bildschirmes.
L	Anfang der letzten Zeile des Bildschirmes.
nG	Go – gehe zur Zeile n. Fehlt die Zeilenangabe n, dann wird zur letzten Zeile der Datei gesprungen.
O	Geht zum Zeilenanfang.

\$	Geht zum Zeilenende.
%	Sucht zur aktuellen Klammer das zugehörige Gegenstück.

Nun folgen die Löschkommandos:

**Tabelle 21: Befehle zum Löschen**

<b>Befehl</b>	<b>Bedeutung</b>
x	Das Zeichen unter dem Cursor löschen.
d\$	Bis zum Zeilenende löschen.
d0	Löscht bis zum Zeilenanfang.
dd	Löscht eine ganze Zeile.
dw	Löscht ein Wort.

Texte verändern die Änderungskommandos:

**Tabelle 22: Änderungsbefehle**

<b>Befehl</b>	<b>Bedeutung</b>
r	Ersetzt das Zeichen unter dem Cursor, ohne in den Eingabe-Modus zu wechseln.
~	Wechselt Groß/Kleinschreibung.
c\$	Ändert bis zum Zeilenende – wechselt in den Eingabe-Modus.
c0	Ändert bis zum Zeilenanfang – wechselt in den Eingabe-Modus.
cc	Ändert die gesamte Zeile – wechselt in den Eingabe-Modus.
cw	Ändert ein Wort.

**Tabelle 23: Suchkommandos**

<b>Befehl</b>	<b>Bedeutung</b>
/string	Sucht – vorwärts nach dem Zeichenmuster, das string entspricht.
?string	Sucht – rückwärts nach dem Zeichenmuster, das string entspricht.
//	Letzten Suchbefehl vorwärts ausführen.
??	Letzten Suchbefehl rückwärts ausführen.

**Tabelle 24: Kopiermöglichkeiten**

<b>Befehl</b>	<b>Bedeutung</b>
yw	Kopiert ein Wort.
yy	Kopiert eine ganze Zeile.
y\$	Kopiert bis zum Zeilenende.
y0	Kopiert bis zum Zeilenanfang.

**Tabelle 25: Diverse Kommandos**

<b>Befehl</b>	<b>Bedeutung</b>
.	Letztes Kommando wiederholen.
J	Join – verbindet zwei Zeilen zu einer.
u	Undo – macht letzten Befehl rückgängig.
U	Aktuelle Zeile wieder herstellen.
:! cmd	Verlässt den vi temporär und führt das Kommando cmd auf der Shell aus.
:r! cmd	Fügt die Ausgabe von cmd an der aktuellen Cursorposition ein.

Mit der optionalen Eingabe einer Zahl *n* kann man die meisten Befehle auch *n*-mal ausführen lassen. Das heißt, wollen wir 6 Zeilen löschen, dann können wir anstelle von 6-maligem Wiederholen des Kommandos `dd` auch kurz `6dd` eingeben. Oder wenn wir 3 Wörter kopieren wollen, können wir kurz `3yw` eingeben.

Wenn wir Dateien in unserem Verzeichnisbaum antreffen, wollen wir aber nicht immer einen Editor starten, um festzustellen, um welche Datei es sich dabei handelt und welchen Inhalt die Datei hat.

Unter Linux gibt es keine Vorgaben oder, besser gesagt, keine Notwendigkeit, den Dateityp durch eine Dateiendung wie `.txt` oder `.doc` zu kennzeichnen. Wenn wir aber keine Dateiendungen haben, kann man auf den ersten Blick nicht erkennen, um welchen Dateityp es sich handelt. Deshalb benutzen wir das Tool ***file***.

```
#file dateiname
```

```
dateiname:UTF-8 Unicode text
```

Bei obiger Datei handelt es sich um eine reine Textdatei und bei dem Beispiel unten um ein Bild im Format GIF mit der Größe 200x128 Pixel.

```
#file datei
```

```
datei: GIF image data, version 89a, 200 x 128
```

Generell möchte ich Sie aber dazu anhalten, auch unter Linux Dateiendungen zu verwenden. Sie müssen nicht der Konvention 8.3 folgen, sondern können sich Ihre eigene zurechtschneiden. Dateiendungen helfen Ihnen, sich schneller einen Überblick zu verschaffen.

Um den Inhalt einer Datei zu betrachten, stehen uns sogenannte ***Pager*** zur Verfügung. Pager sind Programme, die den Inhalt von Dateien in einer komfortablen Art auf dem Bildschirm seitenweise, daher auch der Name, ausgeben können. Die bekanntesten Pager sind ***more*** oder ***less***.

```
#more dateiname
```

**Tabelle 26: Pagersteuerung**

<b>Eingabe</b>	<b>Bedeutung</b>
RETURN	Damit kann man zeilenweise in der Datei weiterscrollen.
CTRL-F oder BILD unten	Damit kann man seitenweise nach unten blättern.
CTRL-B oder BILD rauf	Damit kann man seitenweise nach oben blättern.

Mit dem Befehl **cat**, kann ich mir den gesamten Inhalt einer Datei auf dem Bildschirm ausgeben lassen - allerdings den gesamten Inhalt auf einmal. Ich habe keine Möglichkeit, in der Ausgabe hin und her zu scrollen, wie bei den Pagnern.

```
#cat dateiname
```

### 3.2.4

#### Mitgefangen – Mitgehangen

Es ist endlich an der Zeit, unsere Partition von vorher in den Verzeichnisbaum zu integrieren. Dazu verwenden wir den Befehl **mount**.

Erinnern wir uns daran, dass Dateisysteme unter Linux einfach an ein Verzeichnis gebunden werden. Das bedeutet, wir erstellen zuerst ein Verzeichnis, z. B.: mypart in der ersten Ebene der Verzeichnishierarchie, welches die Partition bzw. das Dateisystem aufnehmen soll.

```
#mkdir /mypart
```

Mit dem Befehl mount kann man nun die Partition an das Verzeichnis binden (einhängen).

Die allgemeine Syntax kann man so beschreiben:

**„Binde was wohin“** also - mount das Gerät /dev/hdb1 an das Verzeichnis /mypart

```
#mount /dev/hdb1 /mypart
```

Nach Bestätigen dieses Befehls können wir auf die erste Partition der zweiten Platte zugreifen, indem wir einfach in das Verzeichnis `/mypart` wechseln.

#### *Beachte*

Das Verzeichnis `mypart` muss vor dem Einhängen bereits existieren. Nach dem Aushängen der Partition ist das Verzeichnis `mypart` nach wie vor vorhanden, aber auf die darauf gespeicherten Daten können wir nicht mehr zugreifen. (Leeres Verzeichnis) Auch werden die Daten, die in einem Verzeichnis gespeichert sind, durch Einhängen (eigentlich Darüberhängen) einer Partition überdeckt. Man kann sich das so vorstellen, als ob die Daten in einem Verzeichnis von der Partition zugedeckt würden. Nach dem Aushängen der überdeckenden Partition sind die ursprünglichen Daten unbeschadet wieder vorhanden.

Dieses System der transparenten Zuschaltung von Plattenplatz gibt uns die Garantie, dass uns eine Festplatte niemals zu klein werden kann. Wenn ich sehe, dass ein Verzeichnis über Gebühr anwächst, kann ich einfach eine neue Platte einhängen, ohne dass ich das gesamte System neu installieren muss.

Der obige `mount`-Befehl ist die einfachste Variante, denn es gibt auch hier eine erkleckliche Anzahl von Optionen.

**Tabelle 27: Optionen für den `mount`-Befehl**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
<code>-a</code>	Bindet alle in der Datei <code>/etc/fstab</code> aufgeführten Dateisysteme ein.
<code>-n</code>	Mountet das Dateisystem ohne einen Eintrag in der Datei <code>/etc/mtab</code> . Notwendig, wenn <code>/etc</code> auf einem nur lesbaren Dateisystem ist.
<code>-r</code>	Bindet das Dateisystem nur lesbar ein.
<code>-w</code>	Bindet das Dateisystem zum Schreiben und Lesen ein. (Defaultwert).
<code>-t type</code>	Angabe des Dateisystemtypes der Partition. Z. B. <code>ext3</code> , <code>xfs</code> ,... Im Normalfall wird das Dateisystem allerdings selbstständig erkannt.



-L Label	Bindet das Dateisystem mit dem angegebenen Label ein.
-v	Verbose-Modus.
-h	Hilfeinformationen.

Um ein CD-ROM Laufwerk, das das erste auf dem zweiten IDE-Controller ist, einzubinden, können wir folgenden Befehl eingeben:

```
#mount -t iso9660 -r /dev/hdc /mnt/cdrom
```

Der **Mountpunkt**, so wird das Verzeichnis genannt, an dem das Dateisystem eingehängt werden soll, /mnt/cdrom ist bei den meisten Distributionen schon nach der Installation automatisch vorhanden. Einige Distributionen haben dafür auch den Mountpunkt (mount point) /cdrom vorgesehen.

Um unser Floppy Disk-Laufwerk in das System zu integrieren, geben wir folgenden Befehl ein:

```
#mount /dev/fd0 /mnt/floppy
```

Auch hier gilt wieder: der Mountpunkt /mnt/floppy oder /floppy ist bereits vorgegeben.

### *Hinweis*

Um eventuellen Missverständnissen zuvor zu kommen: Natürlich bin ich als SuperUser nicht dazu verpflichtet, diese vorgegebenen Mountpunkte zu verwenden, sondern ich kann mir auch, meinen ganz persönlichen Vorlieben entsprechend, Mountpoints selber erstellen – mit welchem Befehl? (mkdir).

Was einmal eingehängt wurde, kann auch wieder ausgehängt werden, und wenn wir uns an den Filesystemcheck erinnern, müssen wir Dateisysteme sogar aushängen, wenn wir sie überprüfen lassen wollen.

```
#umount /dev/hdb1
```

oder

```
#umount /mypart
```

Diese beiden Befehle kann man dazu verwenden, um Dateisysteme aus dem Verzeichnisbaum wieder zu entfernen bzw. auszuhängen. Diese Kurzform kann man wählen, da bei Angabe eines Parameters klar ist, was gemeint ist, denn auf `/mypart` ist eben nur `/dev/hdb1` eingehängt.

**Tabelle 28: Optionen für `umount`**

<b>Option</b>	<b>Bedeutung</b>
-a	Hängt alle in der Datei <code>/etc/mtab</code> aufgeführten Dateisysteme aus. (Ausnahme ist <code>/</code> denn das eine MUSS <sup>4</sup> vorhanden sein – erst im sogenannten Single User Mode kann man dieses Dateisystem aushängen).
-n	Hängt das Dateisystem aus, ohne einen Eintrag in der Datei <code>/etc/mtab</code> vorzunehmen.
-r	Versucht das Dateisystem read only zu remounten, wenn das Aushängen nicht funktioniert hat.
-f	Erzwingt das Aushängen. Bei nicht erreichbaren NFS Systemen.
-v	Verbose-Modus.
-h	Hilfeinformationen.

Nun wollen wir erläutern, welche Bedeutung die schon so oft erwähnten ominösen Dateien `/etc/fstab` und `/etc/mtab` haben.

Die Datei `fstab` enthält Informationen über die Dateisysteme und wird beim Systemstart abgearbeitet. Alle in dieser Datei auf-

---

<sup>4</sup> Muss deshalb, da `/` (ROOT) die Wurzel des Systems repräsentiert. Ohne diesen Bezugspunkt gibt es keinen Ort mehr, wo Programme und Dateien gespeichert werden können.

geführten Dateisysteme werden, bei Vorhandensein, beim Systemstart automatisch, wie in der fstab Datei beschrieben, in das System eingebunden.

Auch der mount-Befehl greift auf diese Datei zu, deshalb kann man nicht nur die Option -a verwenden, sondern auch die bekannte Kurzschreibweise:

```
#mount /mnt/cdrom
```

Durch einen Blick in die fstab weiß das System, welches Gerät wohin gemountet werden soll. Die Reihenfolge der Zeilen ist wichtig, und die Felder in der Datei werden durch Leerzeichen oder Tabulatoren getrennt und haben folgende Bedeutung.

**Tabelle 29: Struktur der fstab-Datei**

<b>Feld</b>	<b>Bedeutung</b>
Erstes Feld	Beschreibt das einzubindende Gerät oder Remote Filesystem (NFS).
Zweites Feld	Gibt den Mountpunkt an. Bei swap-Dateisystemen sollte hier „none“ stehen.
Drittes Feld	Typ des Dateisystemes. (ext2, ext3, reiserfs, ....
Viertes Feld	Mount-Optionen wie z. B. ro, default, .....
Fünftes Feld	Gibt an, ob ein Dateisystem gedumpt (Backup) werden soll oder nicht (0 oder 1).
Letztes Feld	Gibt an, in welcher Reihenfolge beim Systemstart ein Filesystemcheck durchgeführt werden soll. Das / Dateisystem sollte eine 1 haben und alle anderen eine 2. Dateisysteme auf unterschiedlichen Platten werden parallel geprüft, Dateisysteme auf einer Platte sequentiell.

Beispiel einer fstab:

```

/dev/hda1  /          ext3  defaults      1 1
/dev/hda2  /boot      ext3  defaults      1 2
none       /proc       proc  defaults      0 0
/dev/hda3  swap       swap  defaults      0 0
/dev/cdrom  /mnt/cdrom  udf,iso9660 noauto,owner,kudzu ,ro 0 0
/dev/fd0    /mnt/floppy auto    noauto,owner,kudzu 0 0

```

**Tabelle 30: Mögliche Mount-Optionen in der fstab-Datei**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
atime	(Default) Setzt die Zugriffszeit jedesmal neu.
auto	Diese Dateisysteme kann man mit der Option -a mounten.
noauto	Hebt auto auf.
defaults	Default-Optionen: rw, suid, dev, exec, auto, nouser und async.
dev	Interpretiert block- und zeichenorientierte Dateien auf dem Dateisystem.
exec	Erlaubt das Ausführen von Programmen.
nouser	Untersagt es „normalen“ Benutzern, das Dateisystem zu mounten.
ro	Read only.
rw	Read write.
suid	Erlaubt das Setzen von SUID und GID Bits.
user	Erlaubt einem „normalen“ Benutzer, das Dateisystem zu mounten.
users	Erlaubt „normalen“ Benutzern, das Dateisystem zu mounten.

Während die Erstellung und Wartung der Datei `fstab` dem Systemadministrator obliegt, wird die Datei `mtab` nur vom System selbst gewartet und stellt ein Abbild aller Dateisysteme, die wirklich gerade im System eingehängt sind, dar.

*Beispiel einer mtab:*

```
/dev/hda2 / ext3 rw 0 0
none /proc proc rw 0 0
usbdevfs /proc/bus/usb usbdevfs rw 0 0
/dev/hda1 /boot ext3 rw 0 0
none /dev/pts devpts rw,gid=5,mode=620 0 0
none /dev/shm tmpfs rw 0 0
```

Eine ähnliche Ausgabe kann man auch durch das Absetzen des Befehles `mount` ohne Optionen erreichen. Der `mount`-Befehl liest dann die `mtab`-Datei aus und gibt sie etwas anders formatiert wieder aus.

---

# 4

## Die Benutzerverwaltung

---

Zu einer der zentralen Aufgaben eines Systemadministrators zählt die Benutzerverwaltung. Er muss auf seinem System Benutzer anlegen, löschen, modifizieren und ihnen natürlich die passenden Rechte geben können. Das Rechtesystem eines Linux-Computers ist sehr flexibel und von Grund her auf Sicherheit abgestellt.

### 4.1

#### Der Benutzer

Der Benutzer ist die Grundvoraussetzung dafür, dass ein Systemadministrator überhaupt benötigt wird. Oder stellen Sie sich ein System vor, für das Sie zuständig sind und keiner will es – eine erschreckende Vorstellung, nicht wahr?

Wir sollten als angehende Admins ein gutes Mittelmaß aus Pastor und Diktator werden. Ein hervorragendes Klima zwischen Ihnen und Ihren Benutzern hilft auch Ihnen, Ihren Job so gut und so schnell wie möglich zu erledigen. Ansonsten bekommen Sie im Fehlerfall auf die Frage: „Was haben Sie gemacht, als der Fehler auftrat?“ die Standardantwort: „Ich? Nichts!“. Dann können Sie bei Null beginnen und den Fehler suchen. Sie sehen also, es ist immer auch in Ihrem Interesse, wenn Sie Ihre Benutzer wirklich gut behandeln.

Die einfachste Möglichkeit, einen Benutzer auf der Maschine anzulegen, ist mit dem Befehl **useradd**. Wenn wir also einen Benutzer namens `lisi` anlegen wollen, geben wir Folgendes ein:

```
#useradd lisi
```

Ein Benutzer mit dem Namen `lisi`, also der Accountname oder Loginname ist `lisi`, wird auf dem System angelegt. Sie erinnern sich noch, wo das persönliche Heimatverzeichnis des Users ist oder? – Ja im Verzeichnis `/home`, und es heißt so wie der Benutzer. In unserem Fall ist das Home-Verzeichnis also `/home/lisi`.

**Tabelle 31: Optionen für useradd**

<b>Option</b>	<b>Bedeutung</b>
-c	Kommentar oder auch oft als full name des Benutzers bezeichnet.
-d	Damit kann man ein anderes Verzeichnis als /home/benutzername angeben.
-e	Datum, an dem der Account abgeschaltet wird, also der Benutzer ungültig wird.
-f	Gibt die Anzahl der Tage an, ab der der Benutzer gesperrt wird, nachdem sein Passwort abgelaufen ist.
-g	Hier kann man eine spezielle Gruppe per Nummer oder Name angeben. Die Gruppe muss zuvor existieren.
-G	Mit Komma getrennte Liste von Nebengruppen, zu der der Benutzer noch zugehörig ist.
-m	Das HOME-Verzeichnis des Benutzers wird angelegt, falls es nicht existiert, und alle Dateien aus dem Verzeichnis /etc/skel werden in das neue Verzeichnis kopiert. Mit der Option -k kann man auch andere Verzeichnisse als Vorlage angeben.
-n	Bei Red Hat wird defaultmäßig eine Gruppe angelegt, die den gleichen Namen hat wie der Benutzer. Mit dieser Option kann man dieses Verhalten ausschalten.
-p	Das verschlüsselte Passwort. Default ist, den Account zu deaktivieren.
-s	Angabe der Standard-Shell.
-u	Angabe der User-Identifikationsnummer. Diese muss eindeutig sein.

Um festzustellen, mit welchen Defaultwerten das Programm `useradd` arbeitet, kann man `useradd -D` eingeben oder die Datei `/etc/default/useradd` betrachten und nötigenfalls editieren.

```
#useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

Wir wissen, dass die Gruppe, zu der der Benutzer zugehörig sein soll, zuvor existieren muss. Wir müssen also die Gruppe zuerst erzeugen. Dies funktioniert mit dem Kommando **`groupadd`**

```
#groupadd -g 3000 verwaltung
```

Mit diesem Befehl haben wir eine Gruppe mit der Identifikationsnummer (GID) 3000 und dem Namen `verwaltung` angelegt.

**Tabelle 32: Optionen für `groupadd`**

<b>Option</b>	<b>Bedeutung</b>
<code>-g</code>	Eindeutige Gruppennummer.
<code>-r</code>	Weist <code>groupadd</code> an, einen Systemaccount anzulegen – Red Hat-spezifisch.
<code>-f</code>	Diese Option leitet <code>groupadd</code> an, mit einer Fehlermeldung abubrechen, wenn die Gruppe am System bereits existiert.

Die so angelegte Gruppe wird in der Datei **`/etc/group`** gespeichert.

Auszug aus einer Datei `/etc/group`:

```
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
```



```

sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
nfsnobody:x:65534:
mailnull:x:47:
smmisp:x:51:
pcap:x:77:
xfs:x:43:
ntp:x:38:
gdm:x:42:
helmut:x:500:
verwaltung:x:3000:

```

Diese Datei ist in Felder mit dem Feldtrenner : eingeteilt. Wir erkennen den Namen und die Nummer, das x im zweiten Feld würde auf ein Gruppenpasswort hindeuten, von deren Verwendung ich aber persönlich abraten würde.

Will man dennoch ein Gruppenpasswort verwenden, verwendet man dazu den Befehl **gpasswd**. Wir können damit der Gruppe helmut ein Passwort zuordnen. So müssten die User, die in diese Gruppe wechseln wollen, das richtige Passwort kennen.

```

#gpasswd helmut
Neues Kennwort:
Kennwort wiederholen:
#

```

Wenn wir die Datei /etc/group betrachten, hat sich diese überhaupt nicht verändert. Wo steht nun das Passwort?

Alle neuen Linux-Distributionen verwenden das sogenannte shadow-System. Das heißt, das verschlüsselte Passwort steht nicht mehr in der eigentlichen Datei /etc/group, dort wo das x steht, würde eigentlich das verschlüsselte Passwort stehen. Aus Sicherheitsgründen wurde das Passwort aber aus der Datei herausgenommen und in eine eigene Datei umgesiedelt, und zwar in die Datei **/etc/gshadow**.

Betrachten wir einen Auszug aus dieser Datei:

```

root:x::
bin:x::
daemon:x::

```

```
sys:x::
adm:x::
nfsnobody:x::
helmut:4VGdI9082eXte::
verwaltung:x::
```

Wir erkennen, dass bei der Gruppe `helmut` anstelle des `x` nun eine Zeichenkette steht, die das verschlüsselte Passwort repräsentiert.

Nachdem wir alle Optionen für den `useradd`- und `groupadd`-Befehl kennen, wollen wir eine vollständige Kommandozeile für einen Benutzer `maxi` aufbauen, der in der Gruppe `verwaltung` sein soll.

```
#useradd -u 3000 -g 3000 -m -d /home/maximilian -s
/bin/bash -c "Maximilian Baum aus der Buchhaltung" maxi
```

Wenn wir in das Verzeichnis `/home` wechseln, sehen wir, dass wir ein neues Verzeichnis `/home/maximilian` haben. Wechseln wir weiter runter in das Verzeichnis `maximilian` und lassen uns den Inhalt dieses Verzeichnisses mit allen versteckten Dateien anzeigen (mit `ls -a`). Wir erkennen eine große Zahl von Dateien, die mit einem Punkt beginnen (also versteckte Dateien). Diese Dateien sind der Inhalt des Verzeichnisses `/etc/skel`, welche in das neu angelegte HOME-Verzeichnis beim Usererstellen kopiert worden sind. Wenn wir uns den Inhalt des Verzeichnisses von `/etc/skel` mit `ls -a` anzeigen lassen, werden wir erkennen, dass dieser absolut identisch ist.

Wo sind alle anderen Angaben für diesen Benutzer gespeichert?

Die Datei, welche alle benutzerspezifischen Daten aufnimmt, heißt ***passwd*** und ist in dem Verzeichnis ***/etc*** lokalisiert.

Ein Auszug aus der Datei `/etc/passwd`:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
nobody:x:99:99:Nobody:./:/sbin/nologin
sshd:x:74:74:Privilege-separated
SSH:/var/empty/sshd:/sbin/nologin
rpc:x:32:32:Portmapper RPC user:./:/sbin/nologin
```

```
rpcuser:x:29:29:RPC Service
User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS
gdm:x:42:42::/var/gdm:/sbin/nologin
helmut:x:500:500:Helmut Pils:/home/helmut:/bin/bash
user:x:501:501::/mydir:/bin/bash
maxi:x:3000:3000: Maximilian Baum aus der Buchhal-
tung:/home/maximilian:/bin/bash
```

In der letzten Zeile finden wir den User mit den Angaben, die wir gerade beim Anlegen des Users vorgegeben haben.

Sie denken sich nun zu Recht – der Benutzer hat aber noch kein Passwort! Richtig, und deshalb werden wir uns nicht noch länger einer potentiellen Gefahr aussetzen und vergeben mit dem ***passwd***-Befehl sofort ein Passwort für den User *maxi*.

```
#passwd maxi
Changing password for user maxi
New password:
Retype new password:
passwd: All authentication tokens updated successfully.
#
```

### *Einschub*

Passwörter sollten 6 bis 8 Zeichen lang sein und mindestens ein Zeichen aus den folgenden Mengen enthalten:

Klein/Großbuchstaben, Ziffern (0-9), Interpunktionszeichen.

**Tabelle 33: Optionen für *passwd***

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
-g	Damit kann das Passwort für die angegebene Gruppe geändert werden. Gemeinsam mit der Option -r verwendet, wird das Passwort auf die angegebene Gruppe wieder entfernt.
-x	Maximale Gültigkeitsdauer eines Passwortes in Tagen.
-n	Minimale Gültigkeitsdauer in Tagen, Dem Benutzer ist es in dieser Zeit nicht gestattet, sein Passwort zu ändern.

-w	Gibt an, wie viele Tage vor Ablauf des Passwortes der Benutzer gewarnt wird.
-i	Anzahl der Tage, ab der der Account nach Ablauf des Passwortes endgültig gesperrt wird.
-l	Sperrt den Account.
-u	Öffnet einen Account wieder.
-e	Lässt ein Passwort sofort ablaufen, und der Benutzer muss beim nächsten Anmelden das Passwort ändern.
-d	Löscht ein Passwort – Vorsicht!

Sie haben es schon erraten. Wenn wir uns wieder die Datei `/etc/passwd` ansehen, wird sich wieder nichts geändert haben, da wir bei neueren Systemen standardmäßig das Shadow-System verwenden. Somit stehen die verschlüsselten Daten für das Passwort wieder in einer eigenen Datei, und die heißt für die Benutzer ***shadow*** (daher auch der Name).

Ein Blick in die `/etc/shadow`-Datei:

```
root:$1$/rVgXS8m$hsP.W49v4nN3UrOKDeDjN1:12256:0:99999:7:::
bin:!:12256:0:99999:7:::
daemon:!:12256:0:99999:7:::
adm:!:12256:0:99999:7:::
lp:!:12256:0:99999:7:::
sync:!:12256:0:99999:7:::
shutdown:!:12256:0:99999:7:::
rpcuser:!!:12256:0:99999:7:::
nfsnobody:!!:12256:0:99999:7:::
mailnull:!!:12256:0:99999:7:::
smmsp:!!:12256:0:99999:7:::
pcap:!!:12256:0:99999:7:::
xfs:!!:12256:0:99999:7:::
ntp:!!:12256:0:99999:7:::
gdm:!!:12256:0:99999:7:::
helmut:$1$T0511DXb$GQjBUBmnDqyHMDp1AKVc2.:12256:0:99999:7:::
user:!!:12264:0:99999:7:::
```

Die Datei ist wiederum in Felder mit dem Feldtrenner : eingeteilt, wobei diese nicht nur Informationen über das Passwort aufnimmt, sondern auch noch über den Status des Accounts (gesperrt oder offen) und das sogenannte **Passwort-Aging**. Das Passwort-Aging ist dafür zuständig, zu kontrollieren, wann der User sein Passwort ändern muss, wie lange der Account bei abgelaufenem Passwort noch gültig ist, bevor er ganz gesperrt wird u.s.w.

**Tabelle 34: Aufbau der shadow-Datei**

<b>Feld</b>	<b>Bedeutung</b>
1	Login-Name.
2	Verschlüsseltes Passwort.
3	Tage nach dem 1.1.1970, seit dem das Passwort geändert wurde.
4	Tage, nachdem das Passwort geändert werden kann.
5	Tage, nachdem das Passwort geändert werden muss.
6	Anzahl Tage, an denen der Benutzer gewarnt wird, bevor das Passwort abläuft.
7	Anzahl Tage, an denen ein Account noch gültig ist, nachdem das Passwort abgelaufen ist.
8	Tage seit dem 1.1.1970, an dem der Account inaktiv ist.
9	Reserviert.

Die Angaben für das Passwort-Aging können auch über den Befehl **chage** (change aging) verändert werden.

**Tabelle 35: Optionen für chage**

<b>Option</b>	<b>Bedeutung</b>
-m	Minimale Anzahl Tage zwischen Passwortwechsel.
-M	Maximale Anzahl Tage, die ein Passwort gültig ist.
-d	Tage ab 1.1.1970, an dem das Passwort zuletzt geändert wurde.
-E	Account abgelaufen.
-I	Inaktivitätszeit, wo Account noch gültig, obwohl Passwort abgelaufen ist.
-W	Tage der Warnung.

Manchmal kommt es vor, dass ein Mitarbeiter aus einer Firma ausscheidet und man diesen Benutzer aus dem System wieder löschen muss. Das heißt, wir haben einen Account, der gelöscht werden muss, und dies kann mit dem Befehl **userdel** geschehen. Der Befehl `userdel` hat als einzige Option `-r`, mit der man auch das HOME-Verzeichnis des Users löschen kann.

Mit `userdel -r heinrich` ist der Benutzer `heinrich`, inklusive seines Home-Verzeichnisses, vom System verschwunden. War dieser Benutzer noch in einer eigenen Gruppe, dann gehört natürlich auch diese wieder gelöscht und zwar mit dem Befehl **groupdel**.

`groupdel gruppenname`

#### *Hinweis*

Wir dürfen nicht vergessen, dass der user `heinrich` eventuell – wenn er das Recht dazu gehabt hat – auch im restlichen Verzeichnisbaum Dateien angelegt hat und diese nun ohne Besitzer und Gruppenzugehörigkeit herumliegen. Wenn ein anderer Benutzer kommt und zufällig die gleiche ID und GID wie `heinrich` bekommt, auch wenn dies einem anderen Namen entspricht, dann gehören plötzlich diese alten Dateien dem neuen Kollegen. Das ist sicherheitstechnisch nicht gut. Diese Dateien muss man finden und löschen oder kopieren.

Öfter kommt es vor, dass ein Benutzeraccount nur verändert werden soll, weil der Benutzer geheiratet hat oder in eine andere Gruppe gewechselt ist oder, oder, oder. Uns stehen zur Änderung von Einträgen in den Dateien `/etc/passwd` und `/etc/group` spezielle Befehle zur Verfügung und zwar **groupmod** und **usermod**.

**Tabelle 36: Optionen für usermod**

<b>Option</b>	<b>Bedeutung</b>
-c	Kommentar (full name).
-d	HOME-Verzeichnis.
-e	Expire-Datum.
-f	Inactive-Datum.
-g	Primärgruppe.
-G	Sekundärgruppe.
-l	Loginname.
-s	Shell.
-u	UID User-Identifikationsnummer.
-L	Sperrt den Account.
-U	Entsperrt den Account.

**Tabelle 37: Optionen für groupmod**

<b>Option</b>	<b>Bedeutung</b>
-g	GID Gruppen Identifikationsmodus.
-n	Neuer Gruppenname.

Wenn wir allerdings zu einem älteren System kommen, kann es sein, dass noch kein Shadow-System aktiviert wurde. Bei diesem

System stehen dann die Passwörter noch in den Dateien `passwd` und `group`. Damit man zum sichereren Shadow-System wechseln kann, muss man das Passwortsystem konvertieren, was man mit den Befehlen ***pwconv*** und ***grpconv*** realisieren kann. Der Befehl `pwconv` konvertiert das Passwortsystem der Userdaten auf das Shadow-System, und `grpconv` konvertiert das der Gruppen.

Wenn man auf der Kommandozeile

```
#pwconv  
#grpconv
```

eingegeben hat, werden die Passwortsysteme auf das Shadow-System umgestellt.

Sollte man aus irgendwelchen Gründen – mir fällt aber leider überhaupt kein vernünftiger Grund dazu ein – wieder zu einem nicht Shadow-System zurückkehren wollen, dann kann man zurückwechseln, indem man einfach die Befehle ***pwunconv*** und ***grpunconv*** eingibt.

*Hinweis*

Ich möchte noch einmal herausstreichen, dass im Normalfall alle gängigen Distributionen das Shadow-System als Standard vorsehen und installieren.

## 4.2

### Nicht jeder darf alles

Nachdem wir Benutzer auf unserem System haben, müssen wir auch darauf achten, was diese auf unserem System überhaupt für Rechte haben bzw. was sie alles machen dürfen.

Das Rechte-System von Linux baut auf Benutzer und Gruppenrechten auf. Wir haben in den letzten Abschnitten bereits mehrmals einen Blick auf diese Rechte geworfen, ohne näher darauf eingegangen zu sein. Wenn ein Benutzer an einem Linux-System angemeldet ist, ist er unter der Benutzerkennung UID und der Gruppenzugehörigkeit GID dem System bekannt. Das System arbeitet nur mit der UID und der GID. Dass diesen Nummern Namen zugeordnet werden, ist nur ein Kniefall vor uns Usern, da wir Humanoiden besser mit Namen als mit Nummern arbeiten können.

Wenn der Benutzer eine Datei oder ein Verzeichnis anlegt (z. B. mit `touch`, `vi` oder `mkdir`), bekommt dieses Objekt die Kennung, dass es diesem Benutzer (Eigentümer) und dieser Gruppe gehört und welche Rechte (Lesen, Schreiben und Ausführen) der Eigentümer, die Gruppe und schließlich alle anderen Benutzer des Systems für dieses Objekt haben.



Wir sind als root angemeldet und wollen uns dies etwas genauer ansehen. Wir legen dazu ein Verzeichnis in unserem Home-Verzeichnis an, das wir probe nennen. In diesem Verzeichnis wollen wir dann mit dem vi eine Datei mit dem Namen rechte.txt mit dem Inhalt „Hallo Welt“ erstellen.

```
#mkdir /root/probe
#cd /root/probe
#vi rechte.txt
```

Wenn wir uns den Inhalt dieses Verzeichnisses anzeigen lassen mit `ls -la`, erhalten wir folgende Anzeige:

```
insgesamt 12
drwxr-xr-x  2 root  root  4096 31. Jul 14:45 .
drwxr-x--  16 root  root  4096 31. Jul 14:45 ..
-rw-r--r--   1 root  root   199 31. Jul 14:51 rechte.txt
```

Das erste Zeichen steht für den Dateityp. Unter Linux wird, wie wir bereits wissen, alles und jedes als Datei behandelt.

**Tabelle 38: Dateityp-Kennzeichnung**

<b>Zeichen</b>	<b>Bedeutung</b>
d	Directory.
-	Normale Datei.
b	Geräte-Datei für ein blockorientiertes Gerät, z. B. Harddisk.
c	Geräte-Datei für ein zeichenorientiertes Gerät, z. B. Tastatur, Floppy Disk.
l	Steht für einen Link (einen Verweis) auf eine andere Datei.

Die folgenden 9 Zeichenstellen repräsentieren eigentlich Flags für bestimmte Eigenschaften. Die 9 Stellen teilen wir in drei Tripel auf, die die Eigenschaften für Eigentümer, Gruppe und Rest der Welt (also alle anderen Benutzer des Systems) widerspiegeln. Jedes dieser Tripel repräsentiert die Rechte Lesen (r), Schreiben

(w) und Ausführen (x). Für Verzeichnisse bedeutet das x, dass man in dieses Verzeichnis hineinwechseln darf.

So würde z. B. für das erste Tripel die Angabe von rwx bedeuten, dass der Eigentümer auf das Objekt lesend und schreibend zugreifen kann und, wenn es sich um ein Programm handelt, dieses auch ausführen darf (bei einem Verzeichnis bedeutet das x, dass er in dieses Verzeichnis wechseln darf). Die gleichen Angaben werden für die Gruppe und für den Rest der Welt gemacht, und damit kann man sofort ablesen, wer welches Recht auf diesem Objekt besitzt.

Nach der Angabe der Zugriffsrechte kommt der Referenzzähler. Danach kommen die Angaben über den Besitzer (owner) des Objektes und die Gruppe, die im Besitz dieses Objektes ist. Anschließend wird die Größe des Objektes in Bytes angezeigt (außer man verwendet noch die Option -h). Nach dem letzten Änderungsdatum kommt nur noch der Dateiname bzw. der Objektname.

Lassen Sie uns noch einmal einen Blick auf die Zugriffsrechte werfen. Wenn ich eine Zeile wie folgt betrachte: Was kann ich da über die Zugriffsrechte des Objektes, und um welches handelt es sich, sagen?

```
-rwxr-xr-- 1 root root 199 31. Jul 14:51 rechte.txt
```

Dieses Objekt ist eine einfache Datei namens rechte.txt. Die Zugriffsrechte sind folgendermaßen definiert:

- Eigentümer - root  
Lesen, schreiben und ausführen, wenn es ein Programm ist.
- Gruppe - root  
Lesen und ausführen
- Alle anderen  
Nur lesen.

Die Datei ist 199 Bytes groß, wird einmal referenziert und ist am 31. Juli um 14.51 Uhr das letzte Mal verändert worden.

Die Default-Zugriffsrechte, also die nach Erstellen einer Datei angezeigt werden, sind uns vielleicht das eine oder andere Mal nicht genug, da wir mit dieser Datei oder diesem Verzeichnis mehr im Schilde führen. Dann müssen wir die Zugriffsrechte modifizieren.

Wir können die Eigentumsverhältnisse mit **chown** (change owner) verändern.

chown -optionen owner:gruppe objekt

Wenn wir z. B. folgende Datei haben:

```
-rwxr-xr-- 1 root root 4199 31. Jan 11:50 doku.txt
```

und möchten, dass der Eigentümer von nun an der User helmut sein soll, können wir dies durch folgenden Befehl erreichen.

```
#chown helmut doku.txt
```

Möchte ich, dass die Gruppe von doku text auf users geändert werden soll, verwende ich Folgendes:

```
#chown :users doku.txt
```

#### *Hinweis*

Bitte beachten Sie, dass ich hier einen `:` schreiben muss, damit das Kommando users als Gruppenangabe erkennen kann. Natürlich kann ich diese beiden Änderungen auch gleichzeitig vornehmen.

```
#chown helmut:users doku.txt
```

Wäre das Objekt keine einfache Datei, sondern ein Verzeichnis, dann würde nur das Verzeichnis diese neuen Eigentumsverhältnisse bekommen. Sollen aber alle anderen in diesem Verzeichnis enthaltenen Dateien ebenfalls die neuen Einstellungen erhalten, kann man dies mit der Option `-R` für **rekursiv** bewerkstelligen.

Will man nur die Gruppenzugehörigkeit eines Objektes ändern, kann man dies mit dem Befehl **chgrp** erledigen. Unser obiges Beispiel sieht dann folgendermaßen aus:

```
#chgrp users doku.txt
```

Nun wollen wir auch die Zugriffsrechte bzw. den Zugriffsmodus für die drei Bereiche Eigentümer, Gruppe und Rest der Welt ändern. Dazu verwenden wir den Befehl **chmod** (change modus). Das Kommando kennt eine symbolische und eine numerische Art der Zugriffsrechteänderung. Betrachten wir zuerst die symbolische.

Eine Kombination aus den Buchstaben u, g, o, a steuert, wessen (u user, g Gruppe, o Rest der Welt, a alle) Zugriffsrechte geändert werden. Die Zeichen +, -, = verwendet man, um Rechte hinzu, weg oder gleich zu setzen. Mögliche Werte sind r, w oder x. Mehrere symbolische Angaben können durch Komma getrennt werden.

Wir wollen unserer Datei doku.txt andere Zugriffsrechte zuordnen. Wir haben:

```
-rwxr-xr-- 1 helmut users 4199 31. Jan 11:50 doku.txt
```

```
#chmod a+rwx doku.txt
```

```
#ls -l
```

```
-rwxrwxrwx 1 helmut users 4199 31. Jan 11:50 doku.txt
```

Nun wollen wir der Gruppe und dem Rest der Welt das Execute-Recht wieder wegnehmen.

```
#chmod g-x,o-x doku.txt
```

```
#ls -l
```

```
-rwxrwxrwx 1 helmut users 4199 31. Jan 11:50 doku.txt
```

Wir wollen der Gruppe nur Schreib-Rechte geben und dem Rest der Welt Lese- und Schreib-Rechte.

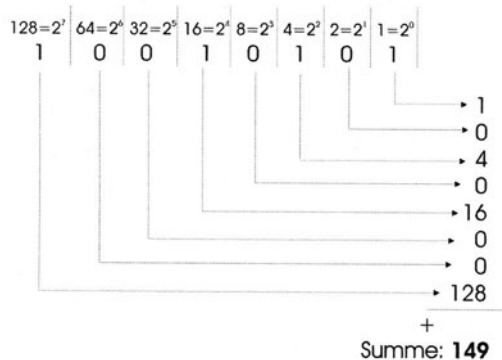
```
#chmod g=w,o=rw doku.txt
```

```
#ls -l
```

```
-rwx-w-rw- 1 helmut users 4199 31. Jan 11:50 doku.txt
```

Wenden wir uns der wichtigen numerischen Art zu. Wenn wir uns die drei Tripel, oder besser noch einen Tripel ansehen, stellen wir fest, dass jede Stelle den Wert eingeschaltet oder ausgeschaltet haben kann. Dies erinnert uns aber schon sehr an das duale Zahlensystem. Wenn wir also die drei Werte mit Dualzahlen repräsentieren, kommen wir auf numerische Werte für jede Kombination der Zugriffsrechte, wobei wir immer ein Tripel alleine betrachten wollen.

*Einschub*



*Abbildung 8: Rechnen im Dualsystem*

*Beispiel*

Betrachten wir die Zugriffsrechte für den Eigentümer:

$r-x$  das bedeutet umgelegt auf unsere Dualschreibweise, dass  $r=1$ ,  $w=0$  und  $x=1$  gesetzt ist. Wir bekommen als Resultat  $(101)_2$ <sup>5</sup>. Umgewandelt in das Dezimalsystem erhalten wir  $(101)_2=5$ .

Genauso gehen wir mit den restlichen zwei Tripel vor und erhalten so eine dreistellige Zahl (eigentlich Oktalzahlen, da der Zahlenbereich von 0 bis 7 geht) Mit diesen so ermittelten Werten gehen wir wieder in das `chmod`-Kommando.

Betrachten wir wieder unser Beispiel von oben:

```
-rwxr-xr-- 1 helmut users 4199 31. Jan 11:50 doku.txt
```

Wir wollen herausfinden, welchen Zahlenwerten die derzeit vergebenen Zugriffsrechte entsprechen:

$$\begin{aligned} rwx &= 1.2^2 + 1.2^1 + 1.2^0 = 4 + 2 + 1 = 7 && \text{für den Eigentümer.} \\ r-x &= 1.2^2 + 0.2^1 + 1.2^0 = 4 + 0 + 1 = 5 && \text{für die Gruppe.} \\ r-- &= 1.2^2 + 0.2^1 + 0.2^0 = 4 + 2 + 1 = 4 && \text{für den Rest der Welt.} \end{aligned}$$

Diese Zugriffsrechte entsprechen also einem Zahlentripel von 754. Die Angabe der Zahlenwerte erfolgt immer in der Reihenfolge Eigentümer, Gruppe und Rest der Welt. Wollen wir der Gruppe alle Rechte geben und dem Rest der Welt alle entziehen, dann geben wir Folgendes ein:

```
#chmod 770 doku.txt
```

Zur Probe lösen wir dies nach obigem Algorithmus, nur in umgekehrter Reihenfolge, wieder auf.

$$\begin{aligned} rwx &= 7 = 4 + 2 + 1 = 1.2^2 + 1.2^1 + 1.2^0 && \text{für den Eigentümer} \\ rwx &= 7 = 4 + 2 + 1 = 1.2^2 + 1.2^1 + 1.2^0 && \text{für die Gruppe} \\ --- &= 0 = 0 + 0 + 0 = 0.2^2 + 0.2^1 + 0.2^0 && \text{für den Rest der Welt} \end{aligned}$$

---

<sup>5</sup> Bei Zahlenangaben, die eine unterschiedliche Interpretation zulassen, in welchem Zahlensystem diese angesiedelt sind, wird die Zahl selbst in Klammern gesetzt und die Basis tiefgestellt angegeben. Also bei Dualsystem 2 bei Hexadezimalsystem 16 u.s.w..

Wollen wir nur dem Eigentümer Lese- und Schreib-Rechte zuteilen, und alle anderen dürfen nichts machen, dann kommt man mit

```
#chmod 600 doku.txt
```

ans Ziel.

Als Option für den **chmod**-Befehl wird vor allem -R für das rekursive Ausführen des Kommandos verwendet. Rekursive Ausführung von Kommandos finden hauptsächlich auf Verzeichnisse Anwendung. Will ich, dass die angegebenen Zugriffsrechte nicht nur für das Verzeichnis gelten, sondern auch für dessen Inhalte und zwar absteigend in alle Unterverzeichnisse, verwendet man die Option -R.

Die Zugriffsrechte fallen natürlich nicht vom Himmel, sondern werden normalerweise vom Sysadmin vorgegeben. Die Vorgabe wird durch den Administrator (SuperUser) des Systems über einen diesen Zweck definierenden Befehl gesteuert. Dieser Befehl heißt **umask**.

Lassen wir uns den aktuellen Wert ausgeben. Dies erreichen wir, indem wir den Befehl umask eintippen:

```
#umask  
0022
```

Wie wirkt diese Dateierstellungsmaske? Wenn man eine Datei erstellt, würde sie per Default die Zugriffsrechte 666 oder rw-rw-rw- bekommen. Dies wird jetzt noch mit der umask abgeglichen, und alles das, was bei der umask gesetzt worden ist, kommt zum Abzug. Wenn wir also unsere umask von 0022 hernehmen, wird überall dort, wo die 2 gesetzt wurde, das ist in diesem Fall die Gruppe und der Rest der Welt, das Recht mit der Repräsentation 2 (das w-Recht) abgezogen, und wenn wir eine Datei erzeugen, bekommen wir automatisch Zugriffsrechte der Form:

```
rw-r--r-- bzw. 644
```

Bei den Directories verhält es sich genauso, nur dass diese standardmäßig mit den Rechten 777 bzw rwxrwxrwx erstellt würden. Ziehen wir wieder die umask ab, dann bekommen wir beim Erstellen eines Verzeichnisses automatisch die Zugriffsrechte 755 für das Verzeichnis zugewiesen.

Jeder Benutzer kann mit dem Befehl umask seine Dateierstellungsmaske ändern. Gesetzt wird die Dateierstellungsmaske durch Angabe jener Werte, die abgezogen werden sollen.

Wir wollen, dass der Rest der Welt per Default keinerlei Rechte auf meine Dateien hat. Dazu verwende ich folgende umask:

```
#umask 0007
```

Jetzt werden sich – so hoffe ich – einige von Ihnen fragen, woher kommt nun plötzlich die 4. Stelle? Haben wir nicht vorher nur von drei Tripel gesprochen, die die Zugriffsrechte festlegen? Woher kommt die Vierte und für was ist sie gut?

Die vierte Stelle, und damit die ganz links außen, kann spezielle Rechte und Eigenschaften auf eine Datei oder ein Verzeichnis legen. Diese Rechte/Eigenschaften sind das **Sticky-Bit**, das **SUID-Bit** (set user ID Bit) und das **SGID-Bit** (set group ID Bit).

**Tabelle 39: Wertetabelle für Sonderrechte**

<b>Wert</b>	<b>Bedeutung</b>
0	Keine speziellen Rechte/Eigenschaften.
1	Setzt das Sticky-Bit.
2	Setzt das SGID-Bit.
4	Setzt das SUID-Bit.

Welche Bedeutung haben die einzelnen zusätzlichen Bits? Wenn man auf einem Verzeichnis das Sticky-Bit setzt, kann jeder in das Verzeichnis schreiben, aber nur der Besitzer kann diese Dateien auch wieder löschen. Gekennzeichnet wird so ein Verzeichnis, indem ganz am Ende der Zugriffsrechte anstelle des x ein t steht.

Ein Beispiel ist das /tmp Verzeichnis.

```
drwxrwxrwt  9 root  root  4096  1. Aug 10:38 tmp
```

SUID und GUID haben folgende Bedeutung. Wir wissen schon, wenn wir am System angemeldet sind und ein Programm aufrufen, läuft dieses Programm auf dem System mit unseren Rechten. Wenn man aber anderen Benutzern etwas erlauben möchte, wofür sie standardmäßig keine Rechte haben, kann man diese beiden Bits verwenden. Diese arbeiten so, dass, wenn z. B. das SUID-Bit gesetzt ist, das Programm beim Aufruf nicht mehr unter dem Rechteschema des Aufrufers abläuft, sondern mit den Rech-

ten dessen, dem das Programm auch gehört. Das SGID-Bit arbeitet genauso, nur dass hier das Rechtesystem der Gruppe zum Tragen kommt. Gekennzeichnet wird dieses Bit im Rechtesystem durch ein *s* anstelle des *x* an der betreffenden Stelle – entweder beim Eigentümer-Tripel für das SUID oder beim Gruppentripel, wenn es um das SGID-Bit geht.

Ein typisches Programm, welches an und für sich nur der *root*-User ausführen darf, ist das *passwd*-Programm zur Vergabe oder Änderung von Benutzerpasswörtern. Nachdem jeder Benutzer sein Passwort ändern können soll, hat dieses Programm das SUID-Bit gesetzt.

```
-r-s--x--x 1 root root 16336 13. Feb 22:19 passwd
```

Dass man auf diese Option besonders Acht geben muss, versteht sich von selbst, wenn man die Systemsicherheit im Auge hat.

Abschließend noch ein Blick auf das Kommando ***chattr***. Damit kann man Dateiattribute auf einem ext2-Dateisystem ändern. Diese Dateiattribute kann man mit ***lsattr*** betrachten.

**Tabelle 40: Dateiattribute**

<b>Kürzel</b>	<b>Attributsname</b>	<b>Bedeutung</b>
a	Append	Daten können nur angehängt werden.
C	Compress	Daten werden automatisch komprimiert.
D	No-dump	Datei wird beim Dumpen ignoriert.
I	Immutable	Datei ist unveränderbar – sogar für <i>root</i> .
S	Secure-delete	Sicheres Löschen durch Überschreiben der Datei mit Nullen.
S	synchronous	Ist die Option <i>sync</i> beim <i>mount</i> -Befehl.
U	Undelete	Löschen kann rückgängig gemacht werden.



Als wichtigste Option sei wieder -R erwähnt, mit der der Befehl wieder rekursiv abgearbeitet wird.

Setzen wir versuchsweise das Dateiattribut für unser Verzeichnis so, dass es automatisch komprimiert wird.

```
#chattr +c /root/probe
#lsattr -d /root/probe
-----c----- ./probe
#lsattr /root/probe
----- ./probe/rechte.txt
```

### 4.3

#### Die Quote

Bei manchen Systemen ist es wichtig, die Menge an Plattenplatz pro Dateisystem, die ein User verbrauchen darf, einzuschränken. So kann man verhindern, dass ein User das Dateisystem völlig für sich selbst in Beschlag nimmt und kein freier Platz mehr für andere Benutzer bestehen bleibt. Unserem Motto „Nicht jeder darf alles“ entsprechend, wollen wir dies beschränken. Dazu verwenden wir den Befehl **quota**. Quotas sind eine Erweiterung des Dateisystems. Zur Vorbereitung von Quotas genügt zunächst eine Änderung der Datei /etc/fstab.

```
/dev/hda1  /      ext2    defaults    1 1
/dev/hda2  swap   swap    defaults    0 0
```

Ändern wir die Datei folgendermaßen um:

```
/dev/hda1  /      ext2    defaults,usrquota,grpquota    1 1
/dev/hda2  swap   swap    defaults          0 0
```

Nun müssen wir die Quotas einrichten

```
# cd /
# touch quota.user
# touch quota.group
# chmod -R 600 quota.user
# chmod -R 600 quota.group
```

Die Quotas werden aktiviert, indem man das Dateisystem neu mountet. Dies können wir mit dem Befehl `mount` von vorhin bewerkstelligen:

```
# mount -n -w -o remount /
```

Damit sind die Quotas aktiv und der Status kann mit dem Befehl ***quotastats*** abgefragt werden.

```
#quotastats
Kernel quota version: 6.5.1
Number of dquot lookups: 0
Number of dquot drops: 0
Number of dquot reads: 0
Number of dquot writes: 0
Number of quotafile syncs: 9
Number of dquot cache hits: 0
Number of allocated dquotes: 0
Number of free dquotes: 0
Number of in use dquot entries (user/group): 0
#
```

Sehr gut - es funktioniert alles!

Damit alles sicher ist, sollten wir eventuell den Rechner doch neu starten. Mit ***quotaon*** und ***quotaoff*** kann man die Quotas ein- und ausschalten.

**Tabelle 41: Optionen für *quotaon*/*quotaoff***

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
-a	Alle Dateisysteme, die in der <code>fstab</code> aufgeführt und mit <code>quotas</code> gekennzeichnet sind, werden eingeschaltet.
-v	Zeigt Informationen über jedes Dateisystem an.
-u	Manipuliert Userquotas (default).
-p	Statusmeldung.
-g	Manipuliert Gruppenquotas.

Das Kommando `edquota -u helmut` startet den Editor.

Quotas for user helmut:

```
/dev/hda3: blocks in use: 0, limits (soft = 1024, hard = 2048)
          inodes in use: 0, limits (soft = 20, hard = 30)
/dev/hda4: blocks in use: 0, limits (soft = 4096, hard = 6144)
          inodes in use: 0, limits (soft = 400, hard = 500)
```

"blocks in use" ist die Gesamtzahl der Blöcke (in Kilobytes), die der Benutzer auf der Partition belegt hat. "inodes in use" ist die Gesamtzahl der Dateieinträge, die der Benutzer auf der Partition angelegt hat.

**Tabelle 42: Optionen für edquota**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
-u	Editiert die Userquotas.
-g	Editiert die Gruppenquotas.
-f FILESYS	Die Änderungen werden nur am spezifizierten Dateisystem FILESYS durchgeführt. Der Defaultwert wäre für alle Dateisysteme mit Quotas.
-t	Editiert die Softlimits.

Mit dem Befehl ***repquota*** kann man einen Report über die Ausnutzung des Dateisystems und die Quotas ausgeben.

`#repquota -av`

\*\*\* Report for user quotas on /dev/hda3 (/var)

Block limits		File limits							
User		used	soft	hard	grace	used	soft	hard	grace
root	--	5615	0	0		563	0	0	
bin	--	2	0	0		3	0	0	
lp	--	1	0	0		1	0	0	
news	--	23	0	0		28	0	0	
uucp	--	18	0	0		17	0	0	
nobody	--	5030	0	0		4160	0	0	
www	--	2	0	0		2	0	0	
kwerner	--	4	2048	4096		1	20	40	

**Tabelle 43: Optionen für repquota**

<b>Option</b>	<b>Bedeutung</b>
-a	Report über alle Dateisysteme, die mit quotas gekennzeichnet sind.
-v	Report über alle Dateisysteme, auch jene, deren quotas nicht benutzt sind.
-t	Schneidet Benutzernamen ab, die länger sind als 9 Zeichen.
-n	Löst UID und GID nicht auf.
-g	Report über Gruppenquotas.
-u	Report über Userquotas.

Eine ähnliche Ausgabe erhält man auch, wenn man den Befehl **quota** verwendet.

```
#quota -ug
```

Damit erhält man Informationen über User- und Gruppenquotas der Dateisysteme.

## 4.4

### Hilfe ist immer und überall

Um den Reigen der am häufigsten benutzten Befehle unter Linux fortzusetzen, ist neben `ls`, `pwd`, `cd` sicherlich auch **man** (manual) zu nennen.

Einer der Gründe, warum Linux als „soooo“ schwer bezeichnet wird, ist die Tatsache, dass die vorhandenen Befehle äußerst komplex sein können. Wir haben schon einige Befehle und deren Optionen kennen gelernt und dabei einen Vorgeschmack darauf erhalten, welche Komplexität in den Befehlen mit ihren Optionen stecken kann. Dass man sich unmöglich alle Befehle mit allen Optionen und Parametern merken kann, wissen auch die Programmierer dieser Tools. Um es einem Benutzer einfacher zu machen, sich Überblick zu verschaffen bzw. Optionen und

Fähigkeiten verschiedener Programme nachlesen zu können, existiert unter Linux ein sogenanntes Hilfesystem mittels Manualseiten. Die Manualseiten sind im Verzeichnis **/usr/share/man** in Sektionen von **man1** bis **man9** in komprimierter Form gespeichert. Mit dem Befehl **man** werden diese Manualseiten dekomprimiert und formatiert am Bildschirm ausgegeben. Versuchen wir dies doch für den uns schon bekannten Befehl **mount**.

```
#man mount
MOUNT(8)                                Linux Programmer's Manual      MOUNT(8)

NAME
    mount - mount a file system

SYNOPSIS
    mount [-lhV]

    mount -a [-fFnrsvw] [-t vfstype] [-O optlist]
    mount [-fnrsvw] [-o options [...]] device | dir
    mount [-fnrsvw] [-t vfstype] [-o options] device dir

DESCRIPTION
    All files accessible in a Unix system are arranged in one big tree,
    the file hierarchy, rooted at /. These files can be spread out over
    several devices. The mount command serves to attach the file system
    found on some device to the big file tree. Conversely, the umount(8)
    command will detach it again.

    The standard form of the mount command, is mount -t type device dir
    This tells the kernel to attach the file system found on device
    (which is of type type) at the directory dir. The previous contents
    (if any)
    :
```

Die Steuerung innerhalb dieser Ausgabe ist so wie in den Pager-Programmen. Mit Return kann man einzelne Zeilen oder mit der Leertaste ganze Bildschirmseiten weiterschalten. Ebenso ist es mit den üblichen Pagerbefehlen möglich, sich in der Ausgabe nach oben oder unten zu bewegen.

Dazu kommt noch die Möglichkeit, so wie im vi mit **/suchwort** ein gewisses Suchwort in der Ausgabe zu finden. Und mit **n** (Next) zum nächsten gefundenen Begriff zu springen. Aus der Ausgabe kommt man mit **q** (quit) wieder heraus.

Zusätzliche Verzeichnisse, an denen die Manpages (die komprimierten Manualseiten) gesucht werden, werden durch die Umgebungsvariable **MANPATH** definiert und festgelegt.

Bei Red Hat liegt das Standardverzeichnis für Manualseiten im Pfad **/usr/share/man** und dort in den Verzeichnissen der jeweiligen Sektion des Manualseiten-Systems, man1 bis man9. Für die erste Sektion ist der Pfad zu den komprimierten manpages also **/usr/share/man/man1**.

**Tabelle 44: Manual-Sektionen**

<b>Sektion</b>	<b>Bedeutung</b>
1	Benutzerkommandos und Programme.
2	Systemaufrufe und Kernelfunktionen.
3	Bibliotheksaufrufe.
4	Spezielle Dateien.
5	Standards und Datei-Formate.
6	Spiele.
7	Makropakete und Konventionen.
8	Befehle für den Systemadministrator.
9	Außergewöhnliche Kernelfunktionen.

Wenn man nach einem Begriff in den Manualseiten sucht, wird sektionsweise gesucht und die **manpage** angezeigt, aus der Sektion, in der der Begriff als erstes gefunden wurde. Wenn man einen Begriff hat, der sowohl für das Programmieren (Sektion 3) als auch für die Systemadministration wichtig ist, bekommt man immer die Hilfeseite für den Programmierer. Damit dies nicht zu einem Problem wird, kann man beim **man**-Befehl die Sektion, in der man suchen möchte, zusätzlich mit angeben.

**Tabelle 45: Optionen für man**

<b>Option</b>	<b>Bedeutung</b>
-D	Setzt das Verhalten von man wieder auf die Defaultwerte zurück.
-M	Alternativer Manualpfad.
-P	Angabe, welcher Pager verwendet werden soll.
-S	Eine mit Doppelpunkt getrennte Liste der zu durchsuchenden Sektionen.
-a	Wenn ein Eintrag gefunden wurde, beendet man das Suchen nicht, sondern sucht in den anderen Abschnitten weiter.
-f	Diese Option ist das Äquivalent zu <i>whatis</i> .
-k	Diese Option ist das Äquivalent zu <i>apropos</i> .
-w	Zeigt nicht die Manualseite, sondern die Lage der Hilfedatei im Verzeichnisbaum an.
-h	Hilfe

```
#man -w mount
/usr/share/man/man8/mount.8.gz
#man -f mount
mount (2) - Aufsetzen und Entfernen von Dateisystemen
mount (2) - mount and unmount filesystems
mount (8) - mount a file system
```

Auch der Befehl ***whatis*** durchsucht die Indexdatenbank nach Einträgen. Mit dem Befehl ***mandb*** kann die Indexdatenbank wieder neu aufgebaut werden.

Mit dem Befehl ***apropos*** wird nach Manualseiten gesucht, die den Suchbegriff beinhalten. Eine weitere Fundgrube für Hilfe und Informationen bietet auch ein eigenes Dokumentationsverzeichnis. Dort sind zusätzliche Informationen zu finden, z. B.: die allseits beliebten und stark genutzten HOWTO's. Dieser Ort ist unter der Red Hat-Distribution ***/usr/share/doc***.

**Tabelle 46: Hilfeffade**

<b>Ort</b>	<b>Bedeutung</b>
/usr/share/doc/FAQ	Frequently Asked Questions.
/usr/share/doc/HOWTO	HOWTO-Dateien.
/usr/share/doc/info	Dokumentationen, die sich der User mit dem Tool info anzeigen lassen kann.

Beim Thema Hilfe darf das Internet und seine Dienste natürlich nicht vergessen werden. Ein erster Einstieg ist hier das Linux Documentation Project:

**<http://www.tldp.org/>**

oder auch

**<http://ldp.ccone.at/HOWTO/HOWTO-INDEX/howtos.html>**

aber auch

**<http://www.linux-docu.de/>**

zu empfehlen.

Einen wahren Schatz und schier unerschöpflichen Fundus an hilfsbereiten Mitmenschen kann man im sogenannten Usenet, also in den Newsgroups finden. Es ist relativ selten, dass man mit seinem Problem keinen kompetenten Gesprächspartner findet, der dieses oder ein ähnliches Anliegen schon lösen konnte.

Laden Sie doch einmal alle Newsgroups mit

comp.os.linux.newbie  
comp.os.linux.answers  
comp.os.linux.hardware  
u.s.w.

Nachdem wir nun wissen, wo wir uns zusätzliche Hilfe besorgen können, wenden wir uns dem Prozessmanagement des Systems zu, damit wir endlich die Shell mit Shell-Skripts voll ausreizen können.



---

# 5

## Prozesse, Prozesse ...

---

... aber wir brauchen dennoch keinen Richter. Wir haben schon einen kurzen Blick unter die Motorhaube geworfen. Nun gehen wir einen Schritt weiter und zerlegen auch noch den Motorblock. In diesem Kapitel werden Sie die grundlegenden Befehle und deren Struktur sowie deren korrekten Ort im Verzeichnisbaum kennen lernen. Als Abschluss und als Vorbereitung einer speziell an unsere Bedürfnisse angepassten Linuxinstallation wird noch der Bootvorgang genauer betrachtet.

### 5.1

#### Prozesse – oder wie behalte ich den Überblick

Wir wissen bereits, dass alle Programme, die auf einem Linux-System gestartet werden, Prozesse heißen. Jeder Prozess (also jedes gestartete Programm) kann für sich selbst weitere Programme, also Prozesse, starten. Diese so gestarteten Prozesse heißen dann Kindprozesse, da Sie von einem anderen Prozess gestartet wurden, und der startende Prozess wird wie heißen? – Natürlich, das ist der Eltern- oder Vaterprozess. Jeder Prozess bekommt eine eindeutige Nummer, mit der er vom System verwaltet und identifiziert wird. Daher auch der Name Prozessidentifikationsnummer **PID**.

Alle Prozesse, die wir bereits in der Shell gestartet haben, sind als Kindprozesse der Shell gelaufen, denn die Shell hat diese Prozesse gestartet. Wir haben auch schon eine Auswirkung davon kennen gelernt. Erinnern Sie sich, wie wir Umgebungsvariablen exportieren mussten, damit die Umgebung bzw. die lokal angelegten Variablen in die Umgebung übergeben wurden und damit den gestarteten Programmen auch zur Verfügung standen.

Um uns einmal einen groben Überblick zu verschaffen, welche Prozesse auf einem System laufen, verwenden wir den Befehl **ps** (process status). Der Befehl **ps** liefert eine Momentaufnahme der aktuellen Prozesse.

**Tabelle 47: Optionen für ps**

<b>Option</b>	<b>Bedeutung</b>
-A	Wählt alle Prozesse aus.
-a	Wählt alle Prozesse mit einem tty aus.
-e	Wählt alle Prozesse aus.
-T	Wählt alle Prozesse dieses Terminals aus.
-r	Wählt nur laufende Prozesse aus.
-s	Wählt nur Prozesse aus, die zu dieser Session gehören.
-u	Auswahl über effektive User-ID.
-l	Ausgabe im Langformat.
-x	Wählt Prozesse ohne kontrollierenden tty aus (Daemons).

#ps -aux

```

USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  1368    84 ?        S    09:20   0:04 init
root         3  0.0  0.0      0     0 ?        SW   09:20   0:00 [kapmd]
root        74  0.0  0.0      0     0 ?        SW   09:21   0:00 [khubd]
root       2716  0.0  0.0  2024    20 ?        S    09:21   0:00 xinetd
root       2764  0.0  0.0  1420   148 ?        S    09:21   0:00 crond
root       2775  0.0  0.3  7612   872 ?        S    09:21   0:00 cupsd
xfs        2834  0.0  0.9  4620  2440 ?        S    09:21   0:00 [xfs]
daemon    2852  0.0  0.0   1412   180 ?        S    09:21   0:00 [atd]
root     3001  0.0  0.3   2536   924 ?        R    09:35   0:00 [fam]
root     3127  0.0  0.5  4480  1332 pts/2    S    09:40   0:00 bash
root     4186  0.0  0.2   2664   720 pts/2    R   14:13   0:00 ps -aux

```

Um aber nicht nur eine Momentaufnahme des Systems, sondern eine kontinuierliche Darstellung zu bekommen, verwenden wir das Tool **top**. Das Verhalten des Programms top kann man zur Laufzeit noch mit folgenden Schaltern beeinflussen:

**Tabelle 48: Optionen für top**

<b>Schalter</b>	<b>Bedeutung</b>
s	Gibt die Zeit in sec. zwischen den Updates an.
p	Beobachtet nur den Prozess mit der beim Aufruf des Programms angegebenen PID (top p 3345).
q	Kontinuierlicher Update.
i	Idle-Prozesse werden nicht angezeigt.
c	Zeigt auch die Kommandozeile der Prozesse an.

#top

```

14:20:03 up 4:59, 4 users, load average: 0.00, 0.00, 0.00
65 processes: 63 sleeping, 2 running, 0 zombie, 0 stopped
CPU states: 2.2% user 0.2% system 0.0% nice 0.0% iowait 97.6% idle
Mem: 255308k av, 249196k used, 6112k free, 0k shrd, 56856k buff
      168780k actv, 0k in_d, 3164k in_c
Swap: 522104k av, 1344k used, 520760k free 86724k cached

```

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	%CPU	%MEM	TIME	CPU	COMMAND
3125	root	15	0	6900	6896	4040	R	1.4	2.7	0:10	0	gnome-termina
3064	root	15	0	4440	4440	988	S	0.8	1.7	0:16	0	Xvnc
3072	root	15	0	1652	1652	760	S	0.2	0.6	0:00	0	twm
1	root	15	0	104	84	56	S	0.0	0.0	0:04	0	init
2	root	15	0	0	0	0	SW	0.0	0.0	0:00	0	keventd
3	root	15	0	0	0	0	SW	0.0	0.0	0:00	0	kapmd
11	root	25	0	0	0	0	SW	0.0	0.0	0:00	0	mdrecoveryd
15	root	15	0	0	0	0	SW	0.0	0.0	0:00	0	kjournald
74	root	25	0	0	0	0	SW	0.0	0.0	0:00	0	khubb

Was bedeuten all diese Felder, die uns der Befehl top anzeigt?

**Tabelle 49: Feldaufschlüsselung für top**

<b>Anzeige</b>	<b>Bedeutung</b>
Uptime	Zeigt die Zeit seit dem letzten Systemstart an und die drei Load Averages, das ist die durchschnittliche Anzahl der Prozesse, die in den letzten 1,5,15 min lauffähig waren. Dies kann man auch mit dem Befehl uptime erhalten.

Prozesses	Totale Anzahl der Prozesse, die beim letzten Update gelaufen sind, unterschieden in running, stopped, sleeping oder untod (oder Zombie).
CPU state	Gibt den prozentualen CPU-Zeitwert in vier Kategorien an. User-mode, system-mode, nice (veränderte Prozesspriorität) und idle.
Mem	Speicherstatistik. Beinhaltet total verfügbaren Speicher, freier Speicher, verbrauchter Speicher, shared memory (für Interprozesskommunikation) und Speicherplatz, der für Buffer verwendet wird.
Swap	Gibt statistische Informationen über den Swap-Bereich aus. Kategorien total, verbraucht, verfügbar. Diese Informationen und auch fürs Memory kann man mit dem Tool free erhalten.
PID	Prozessidentifikationsnummer.
PPID	Elternprozessidentifikationsnummer (parent).
UID	Useridentifikationsnummer.
USER	Der Name des Users, dem der Prozess gehört.
PRI	Priorität des Prozesses.
NI	Nice-Wert. Negative Werte bedeuten höhere Priorität.
SIZE	Die Größe des Prozesses in kB. Eingerechnet werden dabei der Code, die Daten und die Stack-Größe.
TSIZE	Codegröße.
DSIZE	Stack- und Datengröße.
RSS	Die gesamte Größe des Speichers, der von dem Prozess verbraucht wird.

SWAP	Größe der ausgelagerten Teile des Prozesses.
STAT	Der Prozessstatus: S-Sleeping, D-uninterruptable Sleeping, R-Running, T-Stopped und Z-Zombies. Nachgefolgt kann noch die Anzeige < - negativer Wert, N-positiver Nice-Wert, W-swaped out.
TIME	Die gesamte CPU-Zeit.
COMMAND	Das aufrufende Kommando.

Jetzt brauchen wir ein Programm, mit dem wir spielen können – verwenden wir doch einfach den Mozilla. Der Mozilla ist das freie Gegenstück zum Netscape Navigator, und alle Ähnlichkeiten sind nicht rein zufällig!

Das Programm Mozilla heißt `mozilla` und ist im Verzeichnis `/usr/bin` enthalten. Wenn wir `/usr/bin/mozilla` eingeben, dann starten wir den Web-Browser. Vorausgesetzt wird hier natürlich eine grafische Oberfläche.

#### Zwischenfrage

Müssen wir den gesamten Pfad zu dem Programm angeben oder würde die bloße Eingabe des Dateinamens `mozilla` auf der Kommandozeile auch genügen? Wie kann man das feststellen?

Nachdem wir den Prozess `mozilla` gestartet haben und das Bild (ein neues Fenster) aufgegangen ist, merken wir, dass der Prompt in unserem aufrufenden Terminal, also der Shell, nicht wieder erschienen ist. Wir können auch keine Befehle mehr eingeben. Der Prozess, den wir so gestartet haben, läuft im Vordergrund, ist somit ein sogenannter Vordergrundprozess. Wir erkennen die Abhängigkeit dieses Prozesses vom aufrufenden Terminal. Versuchen Sie einmal das Terminal zu schließen. Mit `exit` geht es nicht, da der Kindprozess Mozilla das gesamte Terminal besetzt. Einfach einen Klick in das Schließensymbol. Das Terminal verschwindet und – ja der Mozilla verschwindet mit dem Terminal. Das heißt, ein Prozess ist an ein Terminal gebunden und ist ohne dieses auch nicht „lebensfähig“. Keine Regel ohne Ausnahme. Die sogenannten Daemon-Prozesse haben aber diese Besonderheit, sie sind nicht an ein Terminal bzw. eine Shell gebunden.

Öffnen wir wieder unser Terminal. Es ist nicht optimal, wenn wir für jeden Prozess bzw. für jedes Programm, das wir starten, ein eigenes Terminal öffnen müssten. Es gibt, sie werden es schon gehofft haben, auch für diesen Schönheitsfehler Abhilfe. Wir können nämlich jeden Prozess als sogenannten Hintergrundprozess starten, das Programm also im Hintergrund ablaufen lassen. Das heißt nicht, dass man dies nur mit Automatisierungsprogrammen machen kann, die keine Benutzereingabe benötigen, nein, wir können jegliches interaktive Programm im Hintergrund laufen lassen. Der Kindprozess läuft nur im Hintergrund der Shell, und die Shell kann weitere Aufgaben erledigen. Probieren wir es aus. Der Parameter oder das Zeichen für die Shell, den angegebenen Befehl im Hintergrund zu starten, ist das kaufmännische Und (&) oder auch Amphersant genannt.

```
#mozilla &
[1] 3232
#
```

Der Web-Browser wird wieder geöffnet, und auf der Shell wird in eckigen Klammern eine Zahl, gefolgt von noch einer Zahl ausgegeben. Die Zahl in Klammern bezeichnet die laufende Nummer der im Hintergrund gestarteten Programme. In unserem Fall ist dies nun der erste Job, den wir im Hintergrund laufen lassen. Die zweite Nummer bezeichnet die Prozessidentifikationsnummer. Hier ist dies der Prozess mit der ID von 3232. Diese kann und wird bei Ihnen natürlich eine ganz andere sein.

#### *Hinweis*

Natürlich sind auch Prozesse, die im Hintergrund ablaufen, an die aufrufende Shell gebunden. Einzige Ausnahme sind wieder die Daemons.

Wir sehen in unserer Shell sofort wieder einen Prompt, und wir können ganz normal weiterarbeiten. Geben wir ein paar Befehle ein wie z. B.: `ls -l`, `mount`, `cat /etc/fstab`, ...

Nachdem wir in unserer Arbeit fortführen, haben wir den Hintergrundprozess total vergessen. Um wieder Informationen über unsere Kindprozesse zu erhalten, verwenden wir **jobs**.

**Tabelle 50: Optionen für jobs**

<b>Option</b>	<b>Bedeutung</b>
-l	Listet auch die Prozessnummer mit auf.
-p	Gibt nur die PID aus.
-n	Gibt nur Prozesse aus, die ihren Status geändert haben.
-r	Gibt nur laufende Prozesse aus.
-s	Zeigt nur gestoppte Prozesse an.

```
#jobs
[1]+  Running      mozilla &
#jobs -p
3232
#jobs -l
[1]+  3232  Running      mozilla &
```

Bei diesen Ausgaben bedeutet die Zahl in [ ] die Jobnummer und das + dahinter bedeutet, dass dies der Prozess bzw. Job ist, der als nächstes CPU-Zeit bekommt.

Wir wollen unser Beispiel etwas komplexer werden lassen und starten das Schreibprogramm **oowriter** von OpenOffice.

```
#oowriter &
[2] 3440
```

Beim ersten Aufrufen benötigt der oowriter ein wenig Zeit, da er die Installation durchführen muss. Das Fenster mit der Frage nach einer Datenquelle beenden wir mit einem Klick auf Abbrechen. Danach steht uns ein Word-ähnliches Schreibprogramm zur Verfügung. Was uns aber mehr interessiert, ist die Kommandozeile. Durch das & haben wir den oowriter im Hintergrund gestartet. Wir können wiederum mit unserer Shell weiterarbeiten. Wenn der Prompt nicht gleich wieder erscheint, dann brauchen wir nur ein Return einzugeben und der Prompt ist wieder da. Wir erkennen mit [2], dass dies der zweite Job ist, den wir im Hintergrund laufen lassen, und er hat die Prozess-ID 3440 (Diese Zahl kann bei Ihnen wieder abweichen).

Wenn wir den Befehl `jobs` wie oben eingeben, erhalten wir:

```
#jobs
[1]-  Running      mozilla &
[2]+  Running      oowriter &
#jobs -p
3232
3440
#jobs -l
[1]-  3232      Running      mozilla &
[2]+  3440      Running      oowriter &
```

Wenn uns allerdings nicht immer alle Hintergrundjobs interessieren, sondern z. B. nur der erste, kann man beim `jobs`-Befehl eine Jobspezifikation mit angeben.

```
#jobs -l 1
[1]+  3232  Running      mozilla &
```

Nehmen wir einmal an, dass ich einen Prozess gestartet habe – im Vordergrund – und nun komme ich darauf, dass ich noch etwas anderes machen sollte. Wie immer in der EDV führen auch hier wieder 1000 Wege nach Rom. Betrachten wir einmal diesen. Ich kann jeden im Vordergrund laufenden Job stoppen. Wir kennen den Status `stopped` noch vom `ps` und `top`-Kommando. Einen Vordergrundprozess kann ich mit der Tastenkombination **STRG-z** bzw. **CTRL-z** stoppen.

#### Achtung

Nicht verwechseln mit **STRG-c**, denn diese Tastenkombination beendet ein Shellkommando, und unser Befehlsaufruf ist im Prinzip nichts anderes als ein Befehlsaufruf.

Als Antwort erhalten wir z. B.:

```
[1]+  Stopped      mozilla
#
```

Nun kann ich den Prozess nachträglich in den Hintergrund stellen und zwar mit dem Befehl **bg** (background) und der Angabe der Jobnummer.

```
#bg 1
[1]+  mozilla &
```

Damit haben wir den Mozilla wieder in den Hintergrund gestellt. Wenn ich einen Job im Hintergrund laufen habe, der mir aber so



viel an Leistung des Computers weg nimmt, dass ich nichts mehr vernünftig machen kann, mache ich was? Sie werden es schon wieder erraten haben – ich hole mir den Prozess in den Vordergrund mit dem Befehl **fg** (foreground) und der entsprechenden Jobnummer. Und dort stoppe ich ihn mit STRG-z solange, bis ich wieder willens bin, diesen Job weiter laufen zu lassen. Entweder im Hintergrund mit `bg jobnummer` oder im Vordergrund mit `fg jobnummer`.

Mit all diesen Varianten haben wir stets einen Kindprozess unserer Shell gestartet. Das heißt, unsere Shell ist der Vaterprozess. Mit dem Befehl **exec command** kann man dies nun umgehen. Das Shellkommando **exec** führt ein Kommando mit den angegebenen Argumenten aus, ohne einen Kindprozess zu erzeugen. Der Befehl `exec command` ersetzt die Shell mit `command`. Es wird für `command` kein neuer Prozess gestartet.

Die allgemeine Syntax lautet:

`exec Optionen command arguments`

**Tabelle 51: Optionen für exec**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
-C	Es wird eine leere Umgebung benutzt.
-l	Damit wird ein – als nulltes (!) Argument der Kommandozeile – an das Kommando weitergegeben. Das ist die Art, wie <code>login</code> ein Programm aufruft.

Jetzt wollen wir vielleicht den einen oder anderen Job wieder loswerden bzw. beenden. Entweder weil wir ihn nicht mehr brauchen oder weil er außer Kontrolle geraten ist oder er sich schlichtweg „erhängt“ hat. Wir haben dazu eine große Zahl von Möglichkeiten. Wir können wie vorhin den betreffenden Job in den Vordergrund holen, wenn dieser nicht ohnehin schon im Vordergrund läuft, und ihn dort mit STRG-c beenden. Wir können uns auch das sogenannte **Signal-System** der Interprozesskommunikation zu Nutze machen. Klingt recht kompliziert, ist es aber nicht. Interprozesskommunikation bedeutet nichts anderes, als dass verschiedene Prozesse über einen gewissen Mechanismus miteinander in Kontakt treten und Informationen austau-

schen können. Den Signal-Mechanismus kann man sich wie eine Art Interrupt vorstellen, jedoch nur softwaremäßig. Das heißt, ein Programm schickt einem anderen Programm ein Signal, und durch dieses Signal wird dieser Prozess veranlasst, etwas Bestimmtes zu erledigen. Allerdings ist, wie so oft im Leben, dieses System nur so gut wie sich die Programmierer strickt daran halten. Langer Rede kurzer Sinn: Wenn der Programmierer des betreffenden Programms und damit des Prozesses keine Vorgehensweisen für das zu erhaltende Signal vorgesehen bzw. einprogrammiert hat, kann das Programm auch nicht richtig darauf reagieren. Wieder eine Ausnahme - ein Signal muss unter allen Umständen unterstützt werden, und das ist das Signal mit der Nummer 9 **SIGKILL** – das Notausignal. Mit diesem Signal kann man jedem Prozess quasi den Boden unter den Füßen wegziehen. Es kann hierbei zu Datenverlusten, zu Inkonsistenzen u.v.m. kommen – Also bitte mit Bedacht einsetzen.

Das Programm, das anderen Prozessen diese Signale senden kann, heißt **kill**. Das Kommando ist aber gar nicht so böse wie es sich anhört. Das kill-Kommando verlangt als Option das zu sendende Signal und natürlich die Prozessidentifikationsnummer des Prozesses, der dieses Signal empfangen soll. Die PID bekommen wir z. B. mit `jobs -l` heraus.

Die ersten 15 der 29 Signale (man 7 signal):

**Tabelle 52: Signalarten**

<b>Signal</b>	<b>Wert</b>	<b>Aktion</b>	<b>Bedeutung</b>
SIGHUP	1	A	Verbindung beendet (Aufgehängt).
SIGINT	2	A	Interrupt-Signal von der Tastatur.
SIGQUIT	3	A	Quit-Signal von der Tastatur.
SIGILL	4	A	Falsche Instruktion.
SIGTRAP	5	CG	Überwachung/Stop Punkt.
SIGABRT	6	C	Abbruch.

SIGUNUSED	7	AG	Nicht verwendet.
SIGFPE	8	C	Fließkomma Überschreitung.
SIGKILL	9	AEF	Beendigungssignal.
SIGUSR1	10	A	Benutzerdefiniertes Signal 1.
SIGSEGV	11	C	Ungültige Speicherreferenz.
SIGUSR2	12	A	Benutzerdefiniertes Signal 2.
SIGPIPE	13	A	Schreiben in eine Pipeline ohne Lesen.
SIGALRM	14	A	Zeitsignal von Alarm(1).
SIGTERM	15	A	Beendigungssignal.

Die Zeichen in der Spalte "Aktion" haben folgende Bedeutung:

**Tabelle 53: Bedeutung der Aktionsspalte**

<b>Aktion</b>	<b>Bedeutung</b>
A	Normalerweise wird der Prozess abgebrochen.
B	Normalerweise wird dieses Signal ignoriert.
C	Normalerweise wird ein "dump core" durchgeführt.
D	Normalerweise wird der Prozess gestoppt.
E	Signal kann nicht abgefangen werden.
F	Signal kann nicht ignoriert werden.
G	Kein POSIX.1 konformes Signal.

Wenn wir einen Prozess mit der PID 3440 so beenden wollen, dass dieser noch die Möglichkeit hat, sich ordnungsgemäß zu

beenden, schicken wir ihm das Signal SIGTERM oder 15 mit dem kill-Kommando.

```
#kill -15 3440
```

```
[1]+  Exit 15          mozilla
```

Nur in Notfällen, z. B. wenn der Prozess auf die Signale 15 oder 3 nicht mehr reagiert, kann man das Signal 9 senden.

Eine weitere Möglichkeit, einen Prozess zu beenden, ist nicht über seine PID sondern über seine Jobnummer, die ich über jobs herausbekommen kann. Einziger Unterschied zu den vorherigen Kommandozeilen ist, dass auf die Jobnummer mit dem %-Zeichen Bezug genommen wird.

```
#kill -15 %1
```

Dieses Kommando würde den Job mit der Nummer 1 mit dem Signal SIGTERM beenden.

Wenn man den Befehl **nobup** vor ein Kommando bzw. eine gültige Kommandozeile schreibt, ist dieses Kommando immun gegen das Signal 1 Hangup. Die Ausgabe des Befehls ist nicht standardmäßig der Bildschirm, sondern wird sofort in eine Datei geschrieben. Die Datei befindet sich in jenem Verzeichnis, in dem wir gerade sind, also unser aktuelles Arbeitsverzeichnis, und heißt **nobup.out**.

Wenn ein Prozess gestartet wird, läuft dieser mit einer bestimmten voreingestellten Priorität ab. Negative Werte zeigen höhere Priorität, positive niedrigere Priorität an. Diese Priorität ist für den sogenannten Scheduler wichtig. Der Scheduler hat in einem Multitasking-Betriebssystem wie Linux unter anderem die Aufgabe, den verschiedenen Prozessen nach einem gewissen Regelwerk immer wieder CPU-Zeit zur Verfügung zu stellen und diese auch wieder zu entziehen. Auf diese Priorität kann mit dem Befehl **nice** Einfluss genommen werden. Der Wertebereich für nice liegt von -20 (höchste Priorität) bis +19 (niedrigste Priorität).

```
#nice -n -20 mozilla
```

Damit würde der Web-Browser mozilla mit der höchsten Priorität ablaufen. Der nice-Wert wird im Befehl top angezeigt. Wenn man die vergebene oder die nicht vergebene Priorität eines Prozesses verändern möchte, verwendet man den Befehl **renice**.

Man kann mit `renice` ganze Prozessgruppen oder alle Prozesse eines bestimmten Benutzers anders priorisieren. Im Normalfall wird allerdings der Prozess, den man verändern will, explizit mit seiner PID angesprochen.

```
#renice +2 4456
```

Dieser Befehl ändert die Priorität des Prozess 4456 um 2 ins Positive, also ins Langsamere. Der `nice`-Wert ändert sich.

## 5.2

### Befehle, Befehle oder ein Prozess kommt selten allein

Wir haben beim ersten Kontakt mit den Befehlen `ls`, `cat`, ... schon Prozesse gestartet und die Grundlagen des Dateimanagements kennengelernt. Nun wollen wir das Dateimanagement auf feste Füße stellen und uns ein wenig mehr Befehle dazu ansehen. Mit dem Befehl ***cp*** (copy) kann man eine Datei oder ein Verzeichnis von einem Ort im Verzeichnisbaum an einen anderen Ort kopieren. Wichtig ist, dass man zusätzlich zu den Optionen immer angeben muss, was wohin kopiert werden soll.

**Tabelle 54: Optionen für `cp`**

<b>Option</b>	<b>Bedeutung</b>
-d	Symbolische Links werden nicht verfolgt.
-f	Wenn eine vorhandene Zieldatei nicht geöffnet werden kann, wird sie gelöscht, und es wird noch einmal probiert.
-i	Interaktiv, vor dem Überschreiben wird nachgefragt.
-H	Symbolische Links folgen.
-l	Anstelle einer Kopie wird eine Verknüpfung erstellt
-L	Symbolische Links werden immer verfolgt.
-p	Dateiattribute wenn möglich erhalten.

-P	Quellpfad an Verzeichnis anhängen.
-R	Rekursives Kopieren von Verzeichnissen.
-S	Erzeugt einen symbolischen Link anstelle einer Kopie.
-u	Kopiert nur, wenn die Quelle neuer ist als das Ziel.
-x	Nur auf dem aktuellen Dateisystem bleiben.

Wir wollen die Datei `/etc/passwd` in unser persönliches Verzeichnis kopieren.

```
#cp /etc/passwd /root/passwortedatei
```

Wir müssen nicht mit absoluten Pfaden arbeiten, sondern können auch relative Pfade verwenden. Außerdem wird bei obigem Beispiel die Datei auch gleich umbenannt. Würde der Name der Datei gleich bleiben und wir uns im Verzeichnis `root` als aktuelles Arbeitsverzeichnis befinden, dann könnten wir auch die kurze Version verwenden.

```
#cp /root/passwd .
```

Wenn wir das Verzeichnis `/wichtig` nach `/archiv` kopieren wollen, dann verwenden wir die Option `-R`.

```
#cp -R /wichtig /archiv
```

Auch hier könnten wir wieder mit der Kurzfassung arbeiten.

Mit der Option `-p` werden die Dateiattribute nicht geändert. Das heißt, dass z. B. die Änderungszeit und die Zugriffsrechte nicht verändert werden. Ansonsten würde die neue Datei, also das Ziel, als aktuellen Zeitstempel die Systemzeit erhalten, und die Zugriffsrechte würden wieder auf den Wert, der sich aus den Defaulteinstellungen mit der `umask` ergibt, angelegt.

Mit dem Befehl ***mv*** (move) kann man Dateien oder Verzeichnisse im Verzeichnisbaum verschieben. Man kann `mv` aber auch dazu benutzen, Dateien einfach umzubenennen.

```
#mv /root/mydatei /tmp/testdatei
```

Damit wird die Datei `mydatei` aus dem `SuperUser-HOME`-Verzeichnis in das Verzeichnis `/tmp` geschoben und dabei umbenannt in `testdatei`. Im Unterschied zum Kopieren ist nun die Datei `mydatei` aus dem `/root`-Verzeichnis verschwunden.

**Tabelle 55: Optionen für `mv`**

<b>Option</b>	<b>Bedeutung</b>
<code>-f</code>	Existierende Ziele entfernen und nie fragen.
<code>-i</code>	Interaktiv, vor dem Überschreiben fragen.
<code>-u</code>	Nur neuere Dateien verschieben.
<code>-b</code>	Erstellt nötigenfalls ein Backup.

Wenn ich eine Datei kopiert habe und feststelle, dass ich sie eigentlich nicht mehr benötige, soll sie aus dem Verzeichnisbaum verschwinden. Dazu verwenden wir den Befehl **`rm`** (remove). Dieses Kommando löscht Dateien oder mit der Option `-R` auch Verzeichnisse. Ohne diese Option kann `rm` nur leere Verzeichnisse löschen.

```
#rm rechte.txt
```

würde die Datei `rechte.txt` aus dem aktuellen Verzeichnis löschen.

**Tabelle 56: Optionen für `rm`**

<b>Option</b>	<b>Bedeutung</b>
<code>-f</code>	Ignoriert nicht vorhandene Dateien.
<code>-i</code>	Interaktiv, vor dem Löschen nachfragen.
<code>-r</code> od. <code>-R</code>	Rekursiv, löscht auch Verzeichnisse, die nicht leer sind.

```
#rm -R /mydir
```

Dieser Befehl löscht das nicht leere Verzeichnis /mydir. Um Verzeichnisse zu löschen, steht uns auch ein eigenes kleines Tool zur Verfügung **rmdir** (remove directory).

**Tabelle 57: Optionen für rmdir**

<b>Option</b>	<b>Bedeutung</b>
-p	Parents, Alle angegebenen Unterverzeichnisse werden gelöscht.
--help	Hilfe

Oft ist es nicht wünschenswert, ein und dieselbe Datei einfach unter anderen Namen mehrmals auf einem System zu haben. Alleine unter dem Aspekt, welches dann die aktuellere Datei ist oder wie es mit dem Speicherplatz auf dem Medium aussieht. Aber auch das Thema „Saustall“ auf dem System ist nicht minder zu beachten. Einen möglichen Ausweg aus diesem Dilemma bieten mir die sogenannten **Links** oder **Verweise** auf Dateien. Es gibt grundsätzlich zwei Arten von Verweisen bzw. Links. Symbolische oder Softlinks und Hardlinks. Beide Linkarten werden durch den Befehl **ln** (link) erstellt.

**Tabelle 58: Optionen für ln**

<b>Option</b>	<b>Bedeutung</b>
--backup	Erzeugt Sicherungen von vorhandenen Zieldateien.
-d	Verzeichnisse Hard verknüpfen.
-f	Vorhandene Ziele entfernen.
-i	Nachfragen vor Entfernen vorhandener Ziele.
-s	Erzeugt einen symbolischen bzw. Softlink.
-n	Behandelt einen symbolischen Link beim Ziel als normale Datei.



Die Syntax, die zu merken ist, lautet (**ln** **Optionen** **Original Ziel**). Ich habe dies in Klammern geschrieben, da dies nicht mit der eigentlichen Bezeichnung konform geht. Meine Schreibweise finde ich aber leichter merkbar, denn sie ist so wie beim Kopieren von Dateien (was wohin).

Wir erstellen einen symbolischen Link mit dem Namen ports von der Datei /etc/services in unser HOME-Verzeichnis.

```
#ln -s /etc/services /root/ports
#ls -l /root/ports
lrwxrwxrwx 1 root root 13 2. Aug 13:25 ports -> /etc/services
#
```

Wir erkennen, dass das erste Zeichen ein **l** anstelle eines **-** für eine normale Datei ist. Die Zugriffsrechte sind alle auf 777 gestellt, da der Verweis für alle zugänglich ist, aber der eigentliche Zugriff durch die Zugriffsrechte der eigentlichen Datei erklärt wird. Auf welches Ziel der Verweis ports zeigt, sieht man ganz hinten in der Zeile nach dem Pfeil.

Wenn man nun auf die „Datei“ ports – ist ja eigentlich nur ein Verweis auf die Datei /etc/services – zugreift, greift man auch tatsächlich auf die Originaldatei zu. Wenn man die Datei ports verändert, verändert man die Originaldatei – sofern es die Zugriffsrechte auf die Originaldatei überhaupt zulassen.

Wir erkennen, dass nur durch Angabe der Option **-s** ein symbolischer Link erzeugt wird. Das bedeutet aber, dass ohne Angabe dieser Option ein Hardlink erzeugt wird.

Was verbirgt sich hinter den Bezeichnungen symbolischer Link und Hardlink? Erinnern wir uns, wie eine Datei in den I-Nodes mit den physikalischen Datenblöcken referenziert wird. Ein Hardlink wird dann so repräsentiert, dass er in dem betreffenden Verzeichnis I-Node einfach einen neuen Eintrag mit der I-Node der Originaldatei hinterlegt. Das heißt, der Hardlink verwendet keine eigene I-Node, sondern den der Originaldatei, verbraucht auf diese Weise aber auch keinen zusätzlichen Speicherplatz. Allerdings muss man mit der Einschränkung leben, dass ein Hardlink nur auf einer Partition bzw. im selben Dateisystem existieren kann. Ein Hardlink kann nicht dateisystemübergreifend arbeiten. Ferner ist zu beachten, dass, wenn bei einem Hardlink die Originaldatei gelöscht wird, der Link noch immer die gültige Datei ist,

da ja auf den I-Node und damit auf die physikalischen Blöcke verwiesen wird.

Einen Hardlink erkennt man nicht aufgrund seiner Langformatausgabe mit `ls -l`, da er als reguläre Datei betrachtet wird. Dennoch handelt es sich um ein und die selbe Datei, und Änderungen werden für alle sichtbar.

Im Unterschied dazu arbeitet der symbolische Link dateisystemübergreifend. Mit symbolischen Links kann man Partitionsgrenzen überwinden. Symbolische Links besitzen eigene Einträge in der I-Node und verbrauchen so zusätzlichen Speicherplatz. Zu beachten ist auch, dass, wenn bei einem symbolischen Link die Originaldatei gelöscht wurde, der Link nach wie vor vorhanden ist, aber ins Leere zeigt. Man kann so eine Zahl von kaputten Links am System erhalten, also darauf Bedacht nehmen. Wenn ein solcher „leerer“ Link mit dem `vi` geöffnet und mit Inhalt gefüllt wird, wird auch die Originaldatei wieder erstellt!

Zum Abschluss dieses Abschnittes wenden wir uns noch einem sehr mächtigen Befehl zu, dem ***find***-Befehl. Wie der Name schon sagt, findet der Befehl etwas, und zwar bevorzugt Dateien und Verzeichnisse, die eine bestimmte Eigenschaft erfüllen. Mit diesem Befehl und mit Kombination anderer Befehle, die wir im nächsten Abschnitt kennen lernen werden, sind sehr komfortable Backupstrukturen aufbaubar.

Die allgemeine Syntax lautet:

`find Verzeichnis -Option -Test -Aktion`

Die wichtigsten Optionen sind:

**Tabelle 59: Optionen für `find`**

<b>Option</b>	<b>Bedeutung</b>
<code>-daystart</code>	Misst die Zeiten für <code>-amin</code> , <code>-atime</code> , <code>-cmin</code> , <code>-ctime</code> , <code>-mmin</code> und <code>-mtime</code> Eigenschaften vom Beginn des aktuellen Tages anstelle der letzten 24 Stunden.

-depth	Bearbeitet den Inhalt jedes Verzeichnisses vor dem Verzeichnis selbst.
-xdev	Durchsucht keine Verzeichnisse in anderen Dateisystemen (auf anderen Partitionen).

Bei den Tests gilt, dass sie für einen wahren Wert eine Null liefern. Wenn die Bedingung erfüllt ist, kommt Null heraus. Für die Tests gibt es folgende Varianten (+N Größer als N, -N kleiner als N):

**Tabelle 60: Testmöglichkeiten**

<b>Test</b>	<b>Bedeutung</b>
-atime N	Auf die Datei ist vor N*24 Stunden zugegriffen worden.
-cnewer Referenzdatei	Der Status der Datei wurde vor weniger Zeit verändert, als seit der letzten Veränderung der Referenzdatei vergangen ist.
-ctime N	Der Dateistatus wurde vor N*24 Stunden geändert.
-empty	Die reguläre Datei oder das Verzeichnis ist leer.
-gid N	Die Datei gehört der Gruppe mit der Kennzahl N an.
-mtime N	Der Inhalt der Datei wurde vor N*24 Stunden verändert.
-nouser	Die Datei gehört keinem im System eingetragenen Benutzer.
-nogroup	Die Datei gehört keiner im System angemeldeten Gruppe.

-uid N	Die UID des Eigentümers ist N.
-name Muster	Der Name der Datei passt zu dem Muster. Muster mit Wildcards müssen unter Hochkomma gestellt werden.

Als letztes kann noch eine Aktion anstehen bzw. ausgeführt werden. Als Standardaktion wird `ls` ausgeführt.

**Tabelle 61: Aktionsmöglichkeiten**

<b>Aktion</b>	<b>Bedeutung</b>
-exec Kommando	Führt das Kommando aus.
-print	Gibt den vollständigen Pfadnamen der getesteten Datei auf die Standard-Ausgabe.
-ls	Zeigt die gefundene Datei im Format von 'ls -dils' an.
-fprint Ausgabe-datei	Schreibt den Pfadnamen der getesteten Datei in die Ausgabedatei.

Als Beispiel suchen wir die Datei `ip_local_port_range`, aber wir wissen nicht, wo auf dem System diese Datei abgespeichert ist.

```
#find / -name ip_local_port_range -ls
4234  0 -rw-r--r--  1 root    root    0 Aug  2 14:31
/roc/sys/net/ipv4/ip_local_port_range
```

Für genauere Informationen muss ich Sie auf die Manualseiten von `find` verweisen, denn alle Möglichkeiten des `find`-Kommandos an dieser Stelle zu erörtern, würde den Intentionen dieses Buches widersprechen.

### 5.3. Texte bearbeiten – Jetzt beginnt der Spaß

Jetzt können wir mit geschwellter Brust tönen, dass wir schon einiges an Rüstzeug erhalten haben, um mit dem System ausgiebig spielen zu können. Da unser Spieltrieb nicht zu bremsen ist, kommt in diesem Abschnitt noch der eine oder andere Befehl hinzu, der unsere Spielkiste noch größer macht.

Mit dem Befehl **cat** haben wir uns eine Datei von oben bis unten ansehen können. Damit längere Dateien nicht nur an uns vorüberauschen, haben wir eine Pipe mit **more** gebaut. Wenn mich bei einer Datei nur der Anfang bzw. das Ende interessiert, kann ich mir diese mit dem Befehl **head** oder **tail** anzeigen lassen. Der Befehl **head** gibt den Anfang einer Datei aus und der Befehl **tail** das Ende.

```
#head /etc/passwd
```

Dieser Befehl gibt mir nun defaultmäßig die ersten 10 Zeilen der Datei **passwd** aus.

**Tabelle 62: Optionen für head**

<b>Option</b>	<b>Bedeutung</b>
-c Größe	Gibt die ersten Größe-Bytes aus.
-n Anzahl	Gibt die ersten Anzahl-Zeilen aus (default ist 10).
-v	Verbose. Gibt den Dateinamen davor aus.

Der nachstehende Befehl gibt die letzten 10 Zeilen der Datei **/etc/services** aus:

```
#tail /etc/services
```

**Tabelle 63: Optionen für tail**

<b>Option</b>	<b>Bedeutung</b>
-c Größe	Gibt die letzten Größe-Bytes aus.
-f	Follow – verfolgt die Dateien am Dateiende, wenn sie angefügt werden.
-n Anzahl	Gibt die letzten Anzahl-Zeilen aus (default ist 10).
--pid	In Kombination mit -f verwendet, um tail zu beenden, wenn der Prozess beendet wird.
-s sec	Anzahl Sekunden zwischen den Versuchen.

Wie wir gesehen haben, haben wir mit dem Befehl `cat` die angegebene Datei vollständig von oben nach unten auf dem Bildschirm ausgeben können. Wenn man die Datei von hinten nach vorne, also vom Ende der Datei bis zu deren Anfang vollständig auf dem Bildschirm ausgeben möchte (oder mit den Dateiumleitungszeichen in einen anderen Kanal umleiten möchte), kann man dies mit dem Befehl ***tac*** (also `cat` von hinten gelesen) erreichen.

Wenn wir Dateien entweder mit `tac` oder `cat` ausgeben, könnte uns dabei ja gleich interessieren, wie viele Zeilen diese Datei hat. Um dieses Ergebnis zu erreichen, verwenden wir das Tool ***wc*** (word count). Dieser Befehl gibt die Anzahl von Bytes, Wörter und Zeilen einer Datei aus.

```
#wc /etc/services
```

```
569 2805 19935 /etc/services
```

Die Standard-Ausgabe ist in der Form Zeilen, Wörter und Bytes.

**Tabelle 64: Optionen für wc**

<b>Option</b>	<b>Bedeutung</b>
-c	Gibt die Byte-Anzahl aus.
-m	Gibt die Anzahl der Zeichen (=1Byte) aus.
-l	Gibt die Anzahl der Zeilen mit dem Newline-Zeichen aus.
-L	Gibt die Länge der längsten Zeile aus.
-w	Die Anzahl der Wörter wird ausgegeben.

Wenn uns nur die Anzahl der Wörter in der Datei /etc/services interessiert, erreicht man dies durch folgende Kommandozeile:

```
#wc -w /etc/services
2805 /etc/services
```

Es ist auch möglich, so wie bei anderen Befehlen, das Pipelining zu verwenden. Wir können den Befehl cat mit dem wc-Befehl kombinieren und erhalten als Ausgabe anstatt dem Inhalt der Datei nur eine Zahl und zwar die der Option entsprechende.

```
#cat /etc/services | wc -l
569
```

Dies findet in Shell-Skripts oft Anwendung. Mit dem Befehl **nl** (number line) kann man die Ausgabe mit Zeilennummern versehen.

**Tabelle 65: Optionen für nl**

<b>Option</b>	<b>Bedeutung</b>
-bStyle	Verwendet Style, um die Zeilen zu nummerieren.
-fStyle	Verwendet Style, um Fußzeilen zu nummerieren .
-hStyle	Verwendet Style, um Kopfzeilen zu nummerieren.
-dCC	Verwendet CC, um logische Seiten zu trennen.
-iZahl	Die Nummerierung wird um Zahl inkrementiert.
-nFormat	Fügt Zeilennummern im angegebenen Format ein: Formatangaben sind: ln linksbündig ohne führende Nullen rn rechtsbündig ohne führende Nullen rZ rechtsbündig mit führenden Nullen
-sString	Fügt String nach jeder möglichen Zeilennummer ein.
--help	Hilfe

Mögliche Angaben für Style sind:

**Tabelle 66: Stylemöglichkeiten**

<b>Option</b>	<b>Bedeutung</b>
a	Nummeriert alle Zeilen.
t	Nummeriert nur nichtleere Zeilen.
n	Nummeriert keine Zeilen.



```
#nl -i2 /etc/services
1#
3# Network services, Internet style
5#
. . .
. . .
87# Port Assignments:
89#
91#          0/tcp    Reserved
93#          0/udp    Reserved
95tcpmux    1/tcp    # TCP Port Service Multiplexer
97tcpmux    1/udp    # TCP Port Service Multiplexer
99compressnet 2/tcp    # Management Utility
. . .
. . .
121echo     7/udp    Echo
123#          8/tcp    Unassigned
125#          8/udp    Unassigned
127discard  9/tcp    Discard sink null
129discard  9/udp    Discard sink null
. . .
```

Mit dem Befehl **od** (octal dump) kann man Dateien auch im Octal- oder in anderen Formaten ausgeben lassen.

```
#od /etc/passwd
0000000 067562 072157 074072 030072 030072 071072 067557 035164
0000020 071057 067557 035164 061057 067151 061057 071541 005150
0000040 064542 035156 035170 035061 035061 064542 035156 061057
0000060 067151 027472 064542 027556 060542 064163 062012 062541
0000100 067555 035156 035170 035062 035062 060504 066545 067157
0000120 027472 061163 067151 027472 064542 027556 060542 064163
0000140 066012 035160 035170 035064 035067 071120 067151 064564
. . .
```

**Tabelle 67: Optionen für od**

<b>Option</b>	<b>Bedeutung</b>
-ARADIX	RADIX entscheidet über das Format. Für RADIX sind folgende Angaben gültig: o octal, d vorzeichenbehaftete Dezimalzahl, x hexadezimal, f Fließkommazahl, u vorzeichenlose Dezimalzahl.
-nZAHL	Limitiert die Ausgabe auf die ersten ZAHL-Bytes.
-tFORMAT	Formatangabe für die Ausgabe.
-wWEITE	Gibt WEITE-Bytes pro Ausgabezeile aus.

Da wir gerade Dateien in anderen Formaten ausgegeben haben, sehen wir uns auch gleich den einfachen Text-Formater **fmt** an. Mit diesem Befehl kann man die Ausgabe rudimentär beeinflussen und Zeilen zusammenführen. Sie erinnern sich sicherlich an das „Zeilen zusammenführen“ und die Länge von Zeilen beim vi. Ähnliches können wir auch mit **fmt** erreichen.

**Tabelle 68: Optionen für fmt**

<b>Option</b>	<b>Bedeutung</b>
-pSTRING	Kombiniert Zeilen, die als prefix STRING besitzen.
-s	Teilt lange Zeilen auf.
-t	Die erste und zweite Zeile haben unterschiedliche Bedeutung.
-u	Es wird ein Leerzeichen nach jedem Wort und zwei Leerzeichen nach jeder Zeile eingefügt.
-wZAHL	Gibt die maximale Zeilenlänge an.

Nachdem wir mit `fmt` die Möglichkeit haben, Zeilen zusammenzufügen, wächst in uns vielleicht auch der Wunsch, verschiedene Dateien zusammenzuführen. Dazu gibt es wieder kleine Programme, die dieses bewerkstelligen. Betrachten wir zunächst den Befehl **`join`**. Dieser Befehl gibt die Zeilen aus, die bei beiden angegebenen Dateien identisch sind bzw. deren Inhalt der beziehenden Felder identisch ist. Zum Beispiel machen wir uns schnell zwei Dateien mit folgenden Inhalten:

datei1:

```
a a1
c c1
b b1
```

datei2:

```
a a2
c c2
b b2
```

Mit der Eingabe des Befehls:

```
#join datei1 datei2
```

erhalten wir:

```
a a1 a2
c c1 c2
b b1 b2
```

**Tabelle 69: Optionen für `join`**

<b>Option</b>	<b>Bedeutung</b>
<code>-j FELD</code>	Vergleichsfeld aus <code>datei1</code> und <code>datei2</code> angeben.
<code>-j1 FELD</code>	Vergleichsfeld (FELD) aus <code>datei1</code> angeben.
<code>-j2 FELD</code>	Vergleichsfeld aus <code>datei2</code> angeben.
<code>-t CHAR</code>	Das Zeichen CHAR ist Feldtrennzeichen.
<code>-i</code>	Die Groß/Kleinschreibung wird nicht berücksichtigt.

Der Befehl ***paste*** geht hier noch einen Schritt weiter. Hier müssen gewisse Felder nicht mehr übereinstimmen, um ein Ergebnis zu erhalten, sondern es werden alle korrespondierenden Zeilen zusammengefügt.

Legen wir wieder zwei Dateien `datei1` und `datei2` mit folgendem Inhalt an.

`datei1:`

1  
2

`datei2:`

a  
b  
c

Wenn wir nun das Kommando `#paste datei1 datei2` absetzen, erkennen wir, dass die beiden Dateien zeilenweise zusammengefügt wurden

**Tabelle 70: Optionen für `paste`**

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
-s	Serielle Abarbeitung anstelle einer parallelen. Dann würden wir für unser Beispiel 1 2 a b c erhalten.
-d ,LIST	Damit kann man eine Liste von zu verwendenden Delimiter definieren, wobei sie abwechselnd verwendet werden.

```
#paste -d '$' datei1 datei2 datei1
1%a$1
2%b$2
%c$c
```

Oft stehen wir vor dem Problem, dass wir Dateien auf Datenträgern bekommen, in denen Tabulatoren enthalten sind. Diese Tabulatoren können aber, wie wir alle wissen, auf unterschiedlichen Betriebssystemen unterschiedliche Definitionen haben, was deren Weite betrifft. So ist z. B. bei DOS/Windows der Standard-Tabulator so lang wie 8 Leerzeichen. Bei einem anderen System können es andere Abstände sein bzw. wurde der Tabulator verändert. Wenn wir diese Dateien übernehmen, kann die Formatierung der Datei „flöten“ gehen. Um die Tabulatoren in eine definierte Anzahl von Leerzeichen zu verwandeln, verwenden wir den Befehl **expand**.

**Tabelle 71: Optionen für expand**

<b>Option</b>	<b>Bedeutung</b>
-i	Konvertiert Tabs nicht, wenn er nicht nach einem Leerzeichen kommt.
-t ZAHL	Konvertiert in ZAHL-Leerzeichen anstelle von den 8 Zeichen.
-t LISTE	Wenn mehrere Tabulatorenweiten anzugeben sind, wird die LISTE mit Kommas getrennt angegeben.

Mit dem Befehl **unexpand** kann man Leerzeichen in Tabulatoren umwandeln.

**Tabelle 72: Optionen unexpand**

<b>Option</b>	<b>Bedeutung</b>
-a	Konvertiert alle Leerzeichen anstelle des Ersten.
-t NUMMER	Hier kann man die Anzahl der Leerzeichen angeben

Mit dem Befehl **split** schließlich kann man eine Datei in mehrere kleinere Dateien aufteilen.

**Tabelle 73: Optionen für split**

<b>Option</b>	<b>Bedeutung</b>
-a N	Verwendet N Stellen als suffix der Dateien (Standard ist 2).
-C GRÖSSE	Schreibt wenn möglich GRÖSSE - Bytes pro Ausgabedatei .
-b GRÖSSE	Schreibt GRÖSSE-Bytes pro Ausgabedatei.
-l ZEILEN	Schreibt ZEILEN-Zeilen pro Ausgabedatei.

Stellen wir uns vor, wir hätten ein Verzeichnis voller Dateien und wir suchen jene Dateien, in denen das Wort „Super“ vorkommt. Wir können mit unseren bisher bekannten Befehlen anfangen und würden diejenigen Dateien, in denen das Wort enthalten ist, sicherlich früher oder später finden. Viel leichter und effizienter geht es aber mit dem **grep**-Befehl.

```
#grep Super *
test:Super
#
```

Damit werden alle Dateien nach dem Wort Super durchsucht. Natürlich kann man anstelle des Wildcards \* auch einen Dateinamen angeben.

**Tabelle 74: Optionen für grep**

<b>Option</b>	<b>Bedeutung</b>
-a	Behandelt eine Binärdatei, als ob es eine Textdatei wäre.
-c	Gibt anstelle der normalen Ausgabe nur eine Zahl für die Anzahl der Übereinstimmungen aus.
-e REGEX	Verwendet Regular Expressions.
-H	Gibt für jede Übereinstimmung auch den Dateinamen aus.
-h	Unterdrückt die Dateinamenausgabe.
-i	Ignoriert Groß/Kleinschreibung.
-R oder -r	Rekursive Ausführung.
-v	Invertiert die Bedeutung der Übereinstimmung.

Um Daten in irgendeiner Form, z.B. alphabetisch, zu sortieren kann man den Befehl **sort** verwenden. Der Befehl sort sortiert die Zeilen einer Textdatei.

**Tabelle 75: Optionen für sort**

<b>Option</b>	<b>Bedeutung</b>
-b	Ignoriert führende Leerzeichen.
-f	Ignoriert die Groß/Kleinschreibung.
-d	Dictionary Order, dabei werden nur Leerzeichen und alphanumerische Zeichen berücksichtigt.
-i	Berücksichtigt nur druckbare Zeichen.
-n	Numerische Ordnung.
-r	Kehrt die Sortierung um.

**Tabelle 76: Zusätzliche Optionen für den sort Befehl**

<b>Option</b>	<b>Bedeutung</b>
-c	Überprüft, ob die angegebene Datei bereits sortiert vorliegt.
-m	Verbindet bereits sortierte Dateien und sortiert nicht.
-o FILE	Schreibt das Ergebnis des Sortiervorgangs in die Datei FILE anstelle der Standard-Ausgabe.
-u	Unique erzeugt mit -c eine strikte eindeutige Ordnung.
-z	Beendet jede Zeile mit einem Nullbyte anstelle einem Newline.

Wenn wir eine Datei namen mit folgendem Inhalt haben:

```
Helmut Pils  
Andrea Bacher  
Maximilian Pils  
Terra Nova
```

so erhalten wir mit folgenden Befehlen die Ausgabe:

```
#sort name  
Andrea Bacher  
Helmut Pils  
Maximilian Pils  
Terra Nova
```

Die defaultmäßige Sortierung ist auf das erste Feld gestellt, in unserem Fall eben der Vorname. Wenn wir nach dem ersten, also dem Vornamen, sortieren und innerhalb dieser Sortierung noch nach dem Nachnahmen sortieren wollen, hilft der nachstehende Befehl weiter:



```
#sort +1 +2 name  
Andrea Bacher  
Terra Nova  
Helmut Pils  
Maximilian Pils
```

Soll die Sortierung umgekehrt sein, kommt die Option `-r` ins Spiel.

```
#sort +1 +2 -r name  
Maximilian Pils  
Helmut Pils  
Terra Nova  
Andrea Bacher
```

## 5.4

### Eine kleine Kanalarundfahrt

Viele von den soeben besprochenen Befehlen geben ihr Ergebnis auf der sogenannten Standard-Ausgabe, also dem Bildschirm aus. Es ist langsam an der Zeit, sich mit den verschiedenen Ein/Ausgabekanälen einer UNIX-Umgebung näher zu befassen. Mit diesem Wissen wird es uns auch möglich, diese Kanäle umzuleiten und somit die Ergebnisse in Dateien zu leiten.

Dem grundsätzlichen Verarbeitungsprinzip eines Computers, **EVA** (Eingabe-Verarbeitung-Ausgabe) folgend, benötigen wir einen Weg, mit dem Computer zu kommunizieren. Dazu müssen wir Eingaben vornehmen und bekommen danach eine Ausgabe. Man spricht dabei von Eingabekanal und Ausgabekanal. Standardmäßig wird zur Eingabe die Tastatur verwendet und zur Ausgabe der Bildschirm. Diese Standardkanäle haben deshalb auch eine spezielle Bezeichnung, nämlich ***stdin*** (Standard-Eingabekanal) und ***stdout*** (Standard-Ausgabekanal). Wenn bei der Verarbeitung des eingegebenen Kommandos ein Fehler auftritt, benötigen wir auch noch einen Kanal, der uns die Fehler ausgibt. Dieser Standard-Fehlerkanal (***stderr***) ist ebenfalls der Bildschirm. Die Bezeichnungen `stdin`, `stdout` und `stderr` dienen zur leichteren Verständlichkeit. Der Computer ist ein Computer, und deshalb behandelt er diese Kanäle auch als Nummern.

**Tabelle 77: Kanalnummern**

<b>Nummer</b>	<b>Kanal</b>
0	stdin
1	stdout
2	stderr

Die Kanäle werden auch oft als Streams bezeichnet. Da ich immer von Standardkanälen spreche, liegt die Vermutung nahe, dass es auch andere Kanäle gibt. Und Sie haben mit dieser Vermutung auch recht.

Wir haben bereits gelernt, dass unter Linux alles wie eine Datei behandelt wird. Das heißt, dass jedes Peripheriegerät und angeschlossene Device an einem Linux-Computer mit einer Datei (Pseudodevice) repräsentiert wird. So auch der eben betrachteten Bildschirm und die Tastatur. Die Standardkanäle Tastatur und Bildschirm (eigentlich Konsole bzw. Terminal) sind Dateien. Es sollte auch möglich sein, jedes Device (und damit Devicedateien) als Eingabekanal oder Ausgabekanal zu verwenden. Wenn wir diesen Gedanken einfach weiterspinnen, sollte es auch möglich sein, ganz „normale“ Dateien als Eingabe- und Ausgabekanäle verwenden zu können.

Und genauso ist es! Aber wie werden diese anderen Kanäle zugeordnet? Um die Standardkanäle zu ändern bzw. umzuleiten, verwenden wir die Dateiumleitungszeichen <, >, >>. Mit der Angabe der Kanalnummer können wir festlegen, welcher Kanal wohin umgeleitet werden soll. Wobei es bei Ein- und Ausgabe nicht notwendig ist, da es ohnehin eindeutig ist. Wichtig wird dies beim Fehlerkanal, der die Nummer 2 trägt.

**Tabelle 78: Dateiumleitung**

<b>Zeichen</b>	<b>Bedeutung</b>
< FILE	Die (Standard-)Eingabe wird aus der Datei (Gerät) FILE gelesen.
> FILE	Die (Standard-)Ausgabe wird in die Datei (Gerät) FILE geschrieben.

>> FILE	Die (Standard-)Ausgabe wird an die Datei (Gerät) FILE angehängt.
2> FILE oder 2>> FILE	Der (Standard-)Fehlerkanal wird in die Datei FILE geschrieben oder an die Datei angehängt.

Um ganz korrekt zu sein, müsste bei den ersten Einträgen in der Tabelle auch eine Zahl stehen. Da es bei dieser Art keine Verwechslungen geben kann, benötigt man keine Nummernangabe. Wenn wir einen Befehl absetzen, dessen Ausgabe in der Datei aus und dessen Fehlermeldungen in der Datei fehler gespeichert werden soll, verwenden wir folgendes Kommando:

```
#sort +1 +2 name 1> aus 2> fehler
```

Wenn beide Meldungen, also die Standard-Ausgabe sowie die Fehlermeldungen, in ein und derselben Datei gespeichert werden sollen, verwendet man die Schreibweise **1>&**.

```
#sort +1 +2 name 1>& ausgabe
```

#### Hinweis

Bei dem > Zeichen wird eine Datei ohne Nachfrage erstellt und, wenn es diese Datei bereits gibt, kommentarlos überschrieben. Das heißt, dass eine Datei bei Verwendung von > immer gelöscht und neu angelegt wird, die notwendigen Zugriffsrechte in dem angesprochenen Verzeichnis vorausgesetzt.

Ein Konstrukt, das oft bei Automatisierungen über den Cron-Dienst verwendet wird, ist das Verwerfen aller Meldungen im „Computernirvana“. Dafür gibt es ein eigenes „Gerät“ und zwar das **Null-Device (/dev/null)**.

```
#sort +1 +2 name 1>& /dev/null
```

Nun wird auch die Arbeitsweise von | der Pipe einleuchtender. Es arbeitet einfach so, dass der Eingabekanal des zweiten Befehls mit dem Ausgabekanal des ersten Befehls kurzgeschlossen wird. Man könnte dies auch erreichen, indem wir die Ausgabe des ersten Befehls in eine Datei umleiten und die Eingabe des zweiten Befehls aus eben dieser Datei lesen. Da diese Arbeitsweise sehr oft vorkommt, hat man den Pipe-Mechanismus eingeführt.

Mit diesem Wissen hochgerüstet, können wir die Ausgabe unserer bisher bekannten Befehle und all derjenigen, die wir noch kennen lernen werden, in Dateien sichern oder direkt zu Gerä-

ten schicken. Wir werden darauf bei den Geräte-Dateien noch einmal zurückkommen.

Wir wollen unsere Passwortdatei `/etc/passwd` sortiert ausgeben und in der Datei `sortpass` abspeichern.

```
#sort /etc/passwd > sortpass
#cat sortpass
adm:x:3:4:adm:/var/adm:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
desktop:x:80:80:desktop:/var/lib/menu/kde:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gdm:x:42:42:./var/gdm:/sbin/nologin
.
.
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin
```

Nun interessiert uns vielleicht nicht die gesamte Passwortdatei, sondern nur die angelegten Benutzer. Wir wollen mit einem Befehl erreichen, dass wir nur die Benutzernamen in eine Datei schreiben können. Der Befehl, den wir dazu verwenden, ist der **cut** (von ausschneiden) Befehl. Der Befehl `cut` gibt bestimmte Spalten einer Datei aus. In der folgenden Tabelle bedeutet die Angabe von LISTE eine oder mehrere durch Komma getrennte Nummernangaben. Es ist auch möglich, Bereiche zu definieren, wobei die Felder, Bytes und Zeichen jeweils von 1-M definiert sind. Eine Angabe von -M heißt bis zum M, und N- lässt das Ende offen.

**Tabelle 79: Optionen für cut**

<b>Option</b>	<b>Bedeutung</b>
-b LISTE	Gibt nur diese Bytes aus, wobei die Tabulatoren und backspaces ebenfalls wie normale Zeichen behandelt werden.
-c LISTE	Dasselbe wie bei -b.

-f LISTE	Gibt nur die bezeichneten Felder aus. Felder werden per default mit Tab getrennt.
-d ZEICHEN	Das ZEICHEN fungiert nun als Feldtrenner anstelle von Tab.

Wenn wir nur die Benutzernamen aus der Datei `/etc/passwd` extrahieren wollen, müssen wir zuerst den Feldtrenner ändern, da in der Datei `passwd` keine Tabulatoren sind. In der Datei `passwd` ist der Feldtrenner ein Doppelpunkt. Also ändern wir den Feldtrenner mit der Option `-d` auf den Doppelpunkt. Die Benutzernamen stehen in der Datei `passwd` an erste Stelle, im Feld 1. So können wir mit folgendem Befehl die Benutzernamen extrahieren:

```
#cut -d ':' -f 1 /etc/passwd
```

Entspricht das Ergebnis den Erwartungen, sind wir bereit, dieses Ergebnis in einer Datei zwischenspeichern.

```
#cut -d ':' -f 1 /etc/passwd > temp
```

```
#sort temp > sortpass
```

Wir kennen allerdings schon den Pipe-Mechanismus und können so die beiden Befehle zu einem zusammenfassen, indem wir schreiben.

```
#cut -d ':' -f 1 /etc/passwd | sort
```

Wir haben damit einen neuen Befehl erstellt und zwar einen, der die Benutzernamen eines Systems in alphabetischer Reihenfolge ausgibt. Wenn wir dieses Ergebnis in der Datei `password` abspeichern wollen, dann schreiben wir einfach:

```
#cut -d ':' -f 1 /etc/passwd | sort > sortpass
```

Sehen wir uns dieses Konstrukt – eigentlich unseren neuen Befehl – an. Wir können langsam erahnen, warum Linux den Nimbus des Kryptischen und Komplizierten hat. Allerdings, wenn man dies nicht so oberflächlich betrachtet, erkennt man genau darin eine der Leistungsfähigkeiten des Systems, da mit wenigen Schritten neue Befehle und damit Systemerweiterungen erstellt werden können.

Mit dem Befehl ***uniq*** kann man duplizierte Zeilen aus einer sortierten Datei entfernen.

**Tabelle 80: Optionen für uniq**

<b>Optionen</b>	<b>Bedeutung</b>
-c	Den Zeilen wird die Anzahl des Vorkommens vorangestellt
-d	Es wird nur eine der doppelten Zeilen ausgegeben.
-D	Alle doppelte Zeilen werden ausgegeben. Die Abtrennung geschieht durch Leerzeichen.
-f N	Die ersten N-Felder nicht vergleichen.
-i	Groß/Kleinschreibung nicht berücksichtigen.
-u	Nur einmal vorkommende Zeilen ausgeben.

Wir haben eine Datei test mit folgendem Inhalt:

```

Helmut
Helmut
Helmut
Lisi
Maxi

```

Wenn wir nun mit dem uniq-Befehl etwas experimentieren:

```

#uniq test
Helmut
Lisi
Maxi
#
#uniq -D test
Helmut
Helmut
Helmut

```

```
#  
#uniq -c test  
3 Helmut  
1 Lisi  
1 Maxi  
#
```

Ein weiterer Befehl, der auf die Art und Weise, wie mit Kanälen umgegangen wird, eingeht, ist der **tee**-Befehl. Er kopiert die Standard-Eingabe zur Standard-Ausgabe und in jede der angegebenen Dateien.

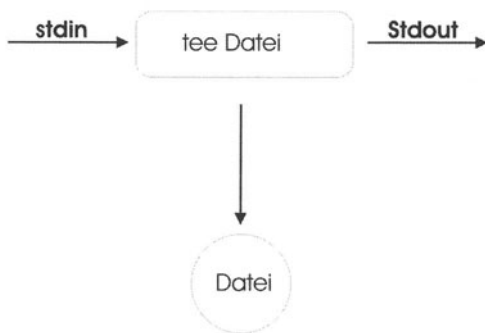


Abbildung 9: Wirkung des tee-Befehls

Tabelle 81: Optionen für tee

Option	Bedeutung
-a	Hängt an die angegebenen Dateien an, anstatt sie zu überschreiben.
-i	Ignoriert Interrupt-Signale.
--help	Hilfeschirm.

Wir können damit unsere obige Absicht der sortierten Benutzerdaten so erweitern, dass wir den nichtsortierten Zwischenschritt mit dem tee-Befehl ebenso in die Datei `unsort` abspeichern und somit für spätere Verwendung zur Verfügung haben.

```
#cut -d ':' -f 1 /etc/passwd | tee unsort | sort > sortpass
```

## 5.5

### Und noch mehr Befehle

Stellen wir uns folgende Situation vor: Wir sind Administratoren eines Web-Servers. Die von uns technisch betreute Web-Site besitzt ein Ausmaß von ca. 700 HTML-Seiten. Nun kommt der Webmaster, also jener, der die HTML-Seiten und den Content betreut, völlig aufgelöst zu uns. Er hat in jeder Datei ein `body`-Tag, der die Farbe des Hintergrundes definiert. Es ist aufgrund einer Änderung des Corporate Designs notwendig, den Hintergrund aller HTML-Dateien auf die neue Farbe zu ändern. Schicken wir den Webmaster mit dem Hinweis, dass er Cascading Stylesheets hätte verwenden sollen, wieder weg, so muss er 700 Seiten editieren. Oder helfen wir ihm mit unserem Wissen weiter?

Natürlich helfen wir ihm! Aber wie? Ein paar Befehle fehlen uns noch zum Glück - also frisch drauf los.

Wenn wir Änderungen in einer Datei durchführen wollen, stellt uns Linux leistungsfähige Tools zur Verfügung. Der Befehl **tr** (translate) macht in diesem Reigen den Anfang. Mit **tr** kann man bestimmte Zeichenketten in einer Datei ersetzen und löschen.

Betrachten wir stellvertretend für unsere 700 HTML-Seiten einmal folgende Datei `seite1.html`:

```
<HTML>
<HEAD>
<TITLE>Das ist die erste Seite</TITLE>
</HEAD>
<BODY bgcolor="#CC00FF">
Hier steht der Text
</BODY>
</HTML>
```

Hier wollen wir alle Vorkommen der Zeichenkette `CC00FF` in die Zeichenkette `CCFFFF` umwandeln.

```
#tr 'CC00FF' 'CCFFFF' < seite1.html
<HTML>
<HEAD>
<TITLE>Das ist die erste Seite</TITLE>
</HEAD>
```



```
<BODY bgcolor="#CCFFFF">
Hier steht der Text
</BODY>
</HTML>
```

Wir erhalten das gewünschte Ergebnis. Die Ausgabe ist allerdings wieder nur am Bildschirm. Wir müssten, um die Datei tatsächlich umzuwandeln, folgende Schritte durchführen:

```
#tr 'CC00FF' 'CCFFFF' < seite1.html > tmp
#mv tmp seite1.html
```

#### *Hinweis*

Geben Sie bitte nicht den folgenden Befehl ein, denn er löscht den Dateiinhalt:

```
#tr 'CC00FF' 'CCFFFF' < seite1.html > seite1.html
```

Der Gedanke hinter diesem Befehl ist sicherlich nahe liegend, und das Gewünschte ist auch realisierbar, allerdings benötigen wir dazu noch ein paar kleine Befehle, die wir am Ende des Kapitels kennen lernen werden.

**Tabelle 82: Optionen für tr**

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
-d	Löscht anstelle von Ersetzen
-s	Ersetzt jede Folge eines wiederholten Zeichens durch nur ein Vorkommen dieses Zeichens.
-t	Schneidet die Menge1 auf die Länge von Menge2 ab, wenn diese angegeben ist.

Die Mengen werden als Zeichenketten angegeben. Die meisten Zeichen stehen für sich selbst.

**Tabelle 83: Zeicheninterpretation**

<b>Zeichenfolge</b>	<b>Interpretation</b>
<code>\NNN</code>	Zeichen mit dem Oktalwert NNN.
<code>\\</code>	Backslash.
<code>\a</code>	Hörbarer Ton.
<code>\b</code>	Zeichen zurück.
<code>\f</code>	Seitenvorschub.
<code>\n</code>	Zeilenvorschub.
<code>\r</code>	Wagenrücklauf.
<code>\t</code>	Horizontaler Tabulator.
<code>\v</code>	Vertikaler Tabulator
<code>Z1-Z2</code>	Von Zeichen Z1 bis Zeichen Z2 aufsteigend.
<code>[:a\num:]</code>	Alle Buchstaben und Ziffern.
<code>[:alpha:]</code>	Alle Buchstaben.
<code>[:cntrl:]</code>	Alle Kontrollzeichen.
<code>[:lower:]</code>	Alle Kleinbuchstaben.
<code>[:upper:]</code>	Alle Großbuchstaben.
<code>[:space:]</code>	Alle horizontalen und vertikalen Leerzeichen und Tabulatoren.

In die gleiche Bresche schlagend wie `tr`, nur ganz anders, ist der **sed** (stream editor). Der Stream Editor wird dazu verwendet, um grundlegende Texttransformationen an einem Eingabestream (`stdin` oder Pipe) durchzuführen. Der `sed` wird durch die Eigenschaft, dass er mit RegEx umgehen kann, sehr mächtig. Was sind nun diese ominösen RegEx (Regular Expressions)? Wir nutzen die Gelegenheit, um einen ersten Blick auf die Regular Expressions zu werfen.

Wie beschreibt man am besten Regular Expression? Regular Expressions kommen eigentlich aus der Mathematik und sind am besten mit Suchmusterübereinstimmung zu beschreiben. Wir alle haben schon mit der Suchmusterübereinstimmung (Pattern Matching) gearbeitet und zwar z. B. beim `ls`-Befehl, als wir mit den Metacharakter `*.jpg` alle Dateien anzeigen lassen wollten, die mit der Endung `.jpg` schließen. Die Regular Expressions gehen nun aber noch einen Schritt weiter und sind wesentlich komplexer. Meiner Meinung nach handelt es sich bei den RegEx um die steinigste Materie, der man beim Erlernen von Linux begegnet.

Mit den RegEx lässt sich sehr schön nach beliebigen Zeichen suchen. Eine Ausnahme bilden Zeichen, die bei den RegEx besondere Bedeutung haben. Diese Zeichen sind: `* + ? . ( ) [ ] { } \ / | ^ $`. Wenn man nach genau diesen Zeichen suchen will, kann man sie genauso wie bei den Shell-Metacharakter mit dem `\` definieren. Wenn wir z. B. nach einem `?` suchen, verwenden wir das Konstrukt `\?`.

**Tabelle 84: RegEx-Ausdrücke**

<b>Zeichen</b>	<b>Bedeutung</b>
<code>.</code>	Passt auf ein beliebiges Zeichen.
<code>*</code>	Passt auf eine beliebige Wiederholung des vorhergehenden Ausdrucks.
<code>+</code>	Eine oder mehrere Wiederholungen des vorhergehenden Ausdrucks.
<code>?</code>	Keine oder eine Wiederholung des vorhergehenden Ausdrucks.
<code>[abcd]</code>	Passt auf genau eines von a oder b oder c oder d.
<code>[^abcd]</code>	Passt auf alles außer auf a oder b oder c oder d.
<code>( )</code>	Gruppieren von Ausdrücken.
<code>abc def</code>	Passt auf abc oder def.
<code>^</code>	Zu Beginn.

\$	Am Ende.
\<	Wortanfang.
\>	Wortende.
\{ \}	Wiederholungsbereiche.

Sehen wir uns ein paar Beispiele an:

Durch den Ausdruck `^0[0-9-]+` findet man 02742-90830-0 aber nicht Wiederholungen wie 882234. Wenn wir nach allen Dateien suchen wollen, die mit `txt` oder `html` aufhören, verwenden wir z. B. `[.](txt|html)$`.

Damit können wir unendlich viele mehr oder weniger komplexe Suchmuster für das Pattern Matching definieren.

Der Stream Editor `sed` kann in seinen Befehlen auch `RegEx` verarbeiten. Diese Eigenschaft macht dieses Tool extrem mächtig. Ein einfaches Suchen und Ersetzen mit dem `sed` funktioniert folgendermaßen:

```
#sed s/HTML/html/g steite2.html
<html>
<HEAD>
<TITLE>Das ist die zweite Seite</TITLE>
</HEAD>
<BODY bgcolor="#CC00FF">
Hier steht noch ein Text
</BODY>
</html>
```

Mit diesem einfachen Befehl haben wir in der Datei `steite2.htm` alle Vorkommen von `HTML` in `html` umgewandelt. Die allgemeine Syntax für einfaches Suchen ist:

```
sed -optionen s/REGEX/ERSATZ/flag dateiname
```

Das `s` steht für `substitute`, danach kommt der Suchstring, der auch über eine `RegEx` definiert sein kann. Es folgt der Ersatzstring und danach ein Flag, das angibt, wo bzw. wie oft diese Ersetzung vorgenommen werden soll. Standardmäßig wird nur jeweils das erste Vorkommen in jeder Zeile ersetzt. Mit einer Zahl kann man angeben, nach dem wievielten Mal des Vorkommens die Ersetzung durchgeführt werden soll.

sed s/HTML/html/5 ersetzt also das 5. Vorkommen des Strings HTML in html. Will man alle Vorkommen ersetzen, verwenden wir das Global-Flag g. Wir können auch einen Bereich definieren, für den die Ersetzung erfolgen soll, also z. B. von Zeile 1 bis 3. In diesem Fall schreiben wir vor das s noch eine Bereichsangabe 1,3s/REGEX/ERSATZ/g. Wenn wir im umgekehrten Fall jedoch eine bestimmte Zeile von der Ersetzung ausnehmen wollen, können wir die Bereichsangabe auch verneinen mit dem Negationszeichen !. Wir schreiben sed 3!s/REGEX/ERSATZ/g, wenn wir die dritte Zeile von der Ersetzung ausnehmen wollen.

**Tabelle 85: Suchoptionen**

<b>Flag</b>	<b>Bedeutung.</b>
g	Global.
p	Zeigt alle Zeilen an, die eine Substitution beinhalten.
w DATEI	Speichert die Substitutionszeilen in der Datei DATEI.
I	Ignoriert Groß/Kleinschreibung.
NUMMER	Ersetzt nur das NUMMER vorkommende Suchergebnis.

**Tabelle 86: Optionen für sed**

<b>Option</b>	<b>Bedeutung</b>
-f FILE	Fügt den Inhalt der Datei FILE zu den auszuführenden Kommandos hinzu.
-e SCRIPT	Fügt SCRIPT zu den auszuführenden Kommandos hinzu.
-r	Verwendet extended regular Expressions.
-s	Behandelt Dateien als separate Dateien und nicht als einen langen kontinuierlichen Stream.

Wir können auch mit dem `sed` unserer Anforderung, dass wir die Farbe der HTML-Dateien ändern, begegnen und zwar mit dem Befehl:

```
#sed s/CC00FF/CCFFFF/g seite2.html > tmp
#mv tmp seite2.html
```

Als Kapitelabschluss betrachten wir noch das mächtige Kommando **xargs**. Mit `xargs` kann man Kommandos bauen und ausführen lassen. Kurz gesagt, erlaubt `xargs` die Weitergabe von Argumenten von einem Kommando zum anderen.

Oft steht man vor dem Dilemma, dass man viele Dateien gleich behandeln möchte. Man möchte etwa alle Dateien, die einem gewissen User gehören, löschen. Das macht man natürlich nicht nur aus Spaß, sondern nur, wenn dieser Mitarbeiter die Firma verlassen hat und die Dateien von ihm nicht mehr benötigt werden. Wir können dieses Ziel mit den Befehlen `find` und `xargs` folgendermaßen lösen:

```
#find / -user seppi -print | xargs rm
```

Hier werden die gefundenen Dateien (inkl. Pfadangabe) mittels der Pipe an den `xargs`-Befehl weitergegeben. Dieser bastelt nun daraus und aus dem `rm` einen eigenen `rm`-Befehl mit allen gefundenen Argumenten. Es wird so nur ein `rm` ausgeführt anstelle von vielen dutzenden `rm`-Befehlen.

Will man die Dateien nur verschieben, dann hilft man sich wie beim `find`-Befehl mit `{ }` aus der Patsche.

```
#find / -user seppi -print | xargs -i mv {} /tmp/backup
```

**Tabelle 87: Optionen für xargs**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
-i	Damit werden die Suchergebnisse in den Befehl eingesetzt. Das Standardzeichen dafür ist <code>{ }</code> .
-0	Die Eingabedateinamen werden durch ein Nullzeichen terminiert und nicht mit einem Leerzeichen.

-p	Interaktive Ausführung.
-t	Gibt die Befehlszeile auf dem Prompt aus, bevor er ausgeführt wird.
-P MAX	Es können MAX-Prozesse gestartet werden. Default ist 1.

Wir sind leider noch immer nicht in der Lage, unserem Kollegen Webmaster in einem Rutsch die Dateien wieder zurecht zu biegen. Denn dazu benötigen wir ein Skript – ein Shell-Skript.

In diesem Kapitel wollen wir den Bootprozess, das Runlevel-Konzept und die beteiligten Dateien von Linux genauer betrachten. Dies ist ein wichtiger Teil, um einen Linux-Computer genau an die Bedürfnisse und Aufgaben, die einem gestellt werden, anzupassen. Wir lernen hier die Leistungsfähigkeit von Linux in der Konfiguration und Anpassbarkeit schätzen.

Wir betrachten exemplarisch den Bootprozess von einer Red Hat-Maschine etwas genauer. Da jeder Linux-Computer für einen bestimmten Einsatzzweck konfiguriert ist, wird auch der Bootprozess bei unterschiedlichen Maschinen unterschiedlich ablaufen – bis auf grundlegende Gemeinsamkeiten.

Am Anfang ist das BIOS – das Basic Input Output System zu nennen. Dieses ist die Aufwärmrunde des PC's. Das BIOS liest die ersten 512 Byte der Festplatte bzw. des Bootdevices ein, wo die eigentlichen Informationen stehen, wo was und wie geladen, also in das RAM transferiert werden soll. In unserem Fall ist dies der Kernel. Das Programm, das den Kernel schließlich lädt, ist der Bootloader. Bei Linux kann dies der **LILLO** (Linux Loader) oder der **GRUB** (Grand Unified Bootloader) sein. Der LILLO ist das am häufigsten verwendete Programm, um Linux zu laden. Allerdings wird der GRUB mit seinen Fähigkeiten zu einer immer stärkeren Konkurrenz zum LILLO. Wir bemerken hier leider wieder eine Zweiteilung zwischen Amerika und Europa. Zum Zeitpunkt, wo dieses Buch geschrieben wurde, war der Stand so, dass Red Hat den GRUB und SuSE den LILLO bevorzugten.

### 6.1

#### Alle ins Boot – der Bootmanager

Der LILLO hat die Fähigkeit, mehrere Betriebssysteme (DOS, Windows, Os/2, ...) zu booten. Der LILLO liest seine Konfigurationsdatei `/etc/lilo.conf` aus und schreibt sich mit diesen Informationen in den Masterbootrecord der Festplatte oder, wenn gewünscht – vor allem in Dualbootkonfigurationen – in den Bootrecord der jeweiligen Partition. Im Serverbetrieb wird man allerdings nie eine Dualbootkonfiguration finden. Dieser Masterbootrecord wird beim Start des Computers nun ausgelesen und



beinhaltet die Informationen, was in das RAM geladen und ausgeführt werden soll. In unserem Fall ist dies der Kernel. Der Kernel ist grundsätzlich für die Erkennung und Unterstützung der gesamten an das System angeschlossenen Hardware zuständig. Wie bereits gesagt, existiert das bekannte Treiberkonzept in der Form von Windows unter Linux nicht. Der letzte Auftrag, den der LILO zu erfüllen hat, ist das Einhängen (Mounten) des ROOT-Filesystems /. Danach wird die Mutter aller Prozesse, der *init*-Prozess, gestartet. Dieser übernimmt den weiteren Startvorgang. Bevor wir uns diesen Startvorgang näher ansehen, betrachten wir eine exemplarische *lilo.conf*.

```
boot = /dev/hda
delay = 40
compact
vga = normal
root = /dev/hda1
read-only
image = /zImage-2.5.99
    label = versuch
image = /zImage-1.0.9
    label = kernel-1.0.9
image = /tamu/vmlinuz
    label = tamu
    root = /dev/hdb2
    vga = ask
other = /dev/hda3
    label = dos
    table = /dev/hda
```

Diese *lilo.conf* bedeutet folgendes:

Der LILO wird angewiesen, den MBR der Platte /dev/hda zu verwenden. Beim Booten wird eine Verzögerungszeit von 40 Dezisekunden gewartet, wo der Benutzer die TAB-Taste drücken kann, um ein Auswahlménü zu erhalten. Im Standardfall sieht man nur die Meldung LILO: am Bildschirm.

Mit der TAB-Taste werden alle Label zu den Images angezeigt und können ausgewählt werden. Wenn man keine Eingabe tätigt, wird das erste Image, das gefunden wird, gestartet. Es kann bis zu 16 verschiedene Images geben. Mit dem Parameter de-

fault = name könnte man einen Kernel bzw. ein Image angeben, das standardmäßig geladen werden soll, das nicht dem ersten Imageeintrag entsprechen muss.

Die ersten 6 Zeilen unserer Konfigurationsdatei gehören zu der sogenannten Global Section.

Der Parameter compact bewirkt, dass Leseaufforderungen an ein und den selben Sektor zusammen ausgeführt werden und so die Ladezeit drastisch reduziert wird.

Der read-only-Parameter weist den LILO an, das ROOT-Filesystem nur lesbar zu mounten. Der root-Parameter definiert jene Partition bzw. jenes Device, welches als ROOT-Filesystem eingebunden werden soll.

Beim vga-Parameter sind folgende Werte zulässig: normal, dies entspricht einem Textmodus von 80x25 und ext, welches einem Textmodus von 80x50 entspricht.

An die Globale-Sektion anschließend findet man die Per-Image-Section. Der image-Parameter enthält die Information, wo das Bootimage, also der zu bootende Kernel, gefunden werden kann. Mit dem label kann man einen Namen vergeben, der bei Betätigung der TAB-Taste angezeigt wird. Der table-Parameter enthält Information, welches Gerät die Partitiontable beinhaltet.

Die letzten drei Einträge in der globalen Sektion gehören zu den sogenannten Kernelparametern und können auch in der Per-Image-Sektion verwendet werden. Sind diese Informationen in der Per-Image-Sektion nicht vorhanden, werden die globalen Informationen verwendet.

Ein wichtiger Parameter ist der append-Parameter. Dieser Parameter kann verwendet werden, um Kernelinformationen mitzugeben, die nicht automatisch ermittelt werden konnten. So kann z. B. die Zeile append = "hd=64,32,202" enthalten sein, wenn die Plattengeometrie nicht erkannt werden kann.

Wenn wir unsere lilo.conf so angepasst haben, wie wir es wollten, führen wir einfach den Befehl **lilo** aus und dieser liest die lilo.conf aus und schreibt diese Informationen in den MBR bzw. in die Partition, die beim boot-Parameter angegeben ist.

**Tabelle 88: Optionen für lilo**

<b>Option</b>	<b>Bedeutung</b>
-v	Verbose
-q	Listet die aktuell gemappten Dateien auf. Lilo verwaltet eine Datei /boot/map, welche den Namen und den Ort des zu bootenden Kernels enthält.
-m FILE	Verwendet FILE anstelle von /boot/map.
-c FILE	Verwendet FILE anstelle von /etc/lilo.conf.
-d DELAY	Verzögerung.
-D LABEL	Name des Default Images.
-t	Testet nur, schreibt nichts.
-i SECTOR	SECTOR wird als Bootsektor verwendet anstelle /boot/boot.b.
-s FILE	Der alte Bootsektor wird gespeichert (/boot/boot.NNN).
-u	Deinstalliert den LILO durch Zurück - kopieren des gesicherten boot sectors.

Nun kann man den Computer neu starten, und alles sollte funktionieren. Wenn einmal Schwierigkeiten beim Starten auftauchen sollten, kann man aufgrund der Fehlermeldungen des LILO schließen, woran es liegt, und eventuell korrigierend eingreifen. Wenn alles gut gegangen ist, wird das Wort LILO angezeigt. Wo bei jeder Buchstabe für eine spezifische Stufe des Ladeprozesses steht.

**Tabelle 89: Fehlercode von lilo**

<b>Fehlermeldung</b>	<b>Bedeutung</b>
Nichts	Es konnte kein Teil des LILO geladen werden. Entweder der LILO ist überhaupt nicht installiert oder es wurde eine falsche Partition gestartet.
L	Die erste Stufe konnte geladen und gestartet werden. Allerdings konnte die zweite Stufe nicht geladen werden (boot.b) – Geometriefehler.
LI	Die zweite Stufe konnte nicht gestartet werden, wurde aber korrekt geladen. Entweder liegt ein Geometriefehler vor oder boot.b wurde verschoben, ohne Lilo aufzurufen.
LIL	Die zweite Stufe wurde gestartet, konnte aber die map-Datei nicht laden. Physikalischer Plattenfehler oder Geometriefehler.
LIL?	Die zweite Stufe des LILO wurde an eine ungültige Adresse geladen. Geometriefehler oder Verschieben von boot.b, ohne lilo aufzurufen.
LIL-	Die Daten in der map-Datei sind ungültig. Geometriefehler oder Verschieben von boot.b, ohne lilo aufzurufen.

Die häufigsten Geometriefehler werden nicht durch physikalische Defekte oder durch ungültige Partitionstabellen verursacht, sondern durch fehlerhafte Installation des LILO wie z. B. die Mißachtung der 1024 Zylindergrenze.

Starke Konkurrenz erwächst dem LILO durch den GRUB. Der GRUB wird nun in einem Unterverzeichnis des /boot-Verzeichnisses konfiguriert. Die Konfigurationsdatei des GRUB heißt **/boot/grub/grub.conf**. Betrachten wir ein Beispiel:

```
default 0
timeout 30
fallback 1
title RED/HAT
root (hd0,0)
kernel /vmlinuz-2.4.20-8 ro root=LABEL=/ hdc=ide-scsi
initrd /initrd-2.4.20-8.img
```

Hier wird eingestellt, dass per default der erste Eintrag gebootet werden soll. Es soll 30 Sekunden gewartet werden, bevor der Bootvorgang weiter geht. Wenn beim Defaultboot irgendetwas schief geht, fällt der Bootvorgang auf den zweiten Eintrag zurück. Der title-Parameter wird beim Starten angezeigt. Die anschließenden Einträge booten dann Red Hat von der ersten Harddisk. Wie wir erkennen können, verwendet der GRUB keinen append-Parameter, um an den Kernel Informationen zu übertragen, sondern sie werden einfach mit Leerzeichen an den kernel-Parameter angefügt.

Will man den GRUB nun installieren, dann verwendet man den Befehl **grub-install** mit der Angabe des Installationsdevices.

```
#grub-install /dev/hda
```

Dieser Befehl installiert den GRUB in den MBR der ersten Harddisk unter Linux.

Wenn man allerdings eine eigene boot-Partition eingerichtet hat, muss man folgenden Befehl verwenden.

```
#grub-install -root-directory=/boot /dev/hda
```

#### *Anmerkung*

Der Befehl grub-install ist eigentlich nur ein Shell-Skript, welches die Grub-Shell **grub** aufruft.

**Tabelle 90: Optionen für die grub-shell**

<b>Option</b>	<b>Bedeutung</b>
--batch	Non-Interaktiv Modus.
--boot-drive	Spezifiziert das stage2 boot device, Default ist (0x0).
--config-file	Spezifiziert stage2 Konfigurationsdatei (Default ist /boot/grub/grub.conf).
--device-map	Verwendet angegebene map-Datei.
--read-only	Schreibt nichts auf das Gerät.
--no-config-file	Verwendet keine Konfigurationsdatei.

Nun haben wir einen der notwendigen Bootmanager installiert und konfiguriert, somit sind wir bereit für den nächsten Schritt – das Runlevelkonzept von Linux.

## 6.2

### Runlevel – bitte nicht davonlaufen

Die Entwicklung von UNIX hat sich in zwei große Zweige geteilt. Einen Berkely Unix (BSD)-Zweig und einen AT&T (System-V)-Zweig. Die beiden Systeme unterscheiden sich unter anderem in der Art und Weise, wie sie booten. Das BSD-System hat wenige große Startdateien, und das System-V-System hat viele kleine Startdateien. Nun habe ich auch schon die Katze aus dem Sack gelassen. Der Bootprozess wird durch Startdateien bestimmt, die entweder eingeschaltet oder ausgeschaltet werden können und somit die Prozesse, die auf einem System laufen, bestimmen.

Linux arbeitet mit einem System-V-Startmechanismus. Dieser Startmechanismus unterscheidet sogenannte Runlevel. Die Runlevel bestimmen, in welchem Status bzw. Modus das System laufen soll. Leider weicht die Definition der Runlevel von Distribution zu Distribution doch oft erheblich ab. Es werden folgende Runlevel unterschieden:

**Tabelle 91: Runlevel unter Red Hat und Suse**

Red Hat:

<b>Runlevel</b>	<b>Bedeutung</b>
0	Halt.
1	Single-User-Mode.
2	Multiuser-Mode ohne NFS.
3	Voller Multi-User-Modus.
4	Frei.
5	X11 Windows-Modus.
6	Reboot.

SuSE:

<b>Runlevel</b>	<b>Bedeutung</b>
0	Halt.
S	Singl-User-Mode.
1	Multiuser-Modus ohne Netzwerk.
2	Multiuser-Modus mit Netzwerk.
3	Multiuser-Modus mit Netzwerk und X11.
4	Frei.
5	Frei.
6	Reboot.

Was sich hinter den einzelnen Runlevels verbirgt und wie sie definiert sind, wird in der Datei **/etc/inittab** festgelegt und konfiguriert. In dieser Datei ist auch obige Tabelle wiedergegeben, damit man nachlesen kann, welche Zuordnung auf dem jeweiligen System gilt.

Betrachten wir exemplarisch einen Ausschnitt der `inittab` von einem Red Hat-System:

```
#
# inittabThis file describes how the INIT process should set up
#         the system in a certain run-level.
#
#   Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#  0 - halt (Do NOT set initdefault to this)
#  1 - Single user mode
#  2 - Multiuser, without NFS
#  3 - Full multiuser mode
#  4 - unused
#  5 - X11
#  6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm -nodaemon
```

In dieser Datei ist das Kommentarzeichen die #. Alles was hinter einem # Zeichen steht, dient nur dem Kommentar und wird nicht



berücksichtigt. Jede Zeile in dieser Datei setzt sich in folgender Form zusammen:

ID:RUNLEVEL:AKTION:PROZESS

Mit ID bestimmt man eine eindeutige Identifikation des Prozesses. Mit RUNLEVEL definiert man, in welchem Runlevel der Prozess aktiviert wird. Im dritten Feld wird definiert, was der init-Befehl mit dem Prozess durchführen soll.

Mögliche Werte im AKTION-Feld.

**Tabelle 92: Aktionsfeld**

<i><b>Aktion</b></i>	<i><b>Bedeutung</b></i>
sysinit	Aktiviert den Befehl nur einmal und zwar nach dem Systemstart.
once	Aktiviert den Befehl, ohne auf dessen Beendigung zu warten.
wait	Aktiviert den Befehl und wartet auf dessen Beendigung.
ctrlaltdel	Aktiviert den Befehl wenn STRG-ALT-DEL gemeinsam gedrückt werden.
respawn	Aktiviert den Befehl beim Übergang in einen der angegebenen Runlevel und startet ihn wieder neu, wenn er beendet wird.

Mit dem Eintrag `id:5:initdefault:` wird der Defaultrunlevel als der Modus definiert, in den das System booten soll, wenn nichts anderes angegeben ist. In unserem Fall ist dies der Runlevel 5 – die grafische Oberfläche X11 Windows-System.

Mit `si::sysinit:/etc/rc.d/rc.sysinit` wird das System initialisiert.

Mit der Information über den Defaultrunlevel – bei uns 5 – sucht der init-Befehl in der `inittab` den folgenden Eintrag:

`15:5:wait:/etc/rc.d/rc 5`

Dieser Eintrag beauftragt den init-Prozess, in das Verzeichnis `/etc/rc.d/rc5.d/` zu wechseln. Dort stehen all jene Startdateien (Start-Skripts), die den Runlevel 5 definieren. Also kann man durch Hinzufügen und Wegnehmen von Start-Skripten in diesem Verzeichnis genau festlegen, wie mein eigenes spezielles System

den Runlevel 5 definiert, und dieser kann weit von dem abweichen, was mein Nachbar hat. Die Start-Skripte haben einen gewissen einzuhaltenden Namensteil. Der Anfang des Dateinamens ist wichtig. Die Dateien in diesem Verzeichnis beginnen entweder mit **S** oder **K**, gefolgt von einer zweistelligen Zahl. S88Web-Server könnte z. B. in diesem Verzeichnis stehen. Das S bedeutet, es handelt sich dabei um ein Start-Skript, und die Zahl definiert die Reihenfolge, in der die Skripte ausgeführt werden. Die Start-Skripte sind dafür zuständig, alle Prozesse, die diesen Runlevel definieren, zu starten. Alles was dahinter kommt, ist beliebig und hat auf die Ausführung keinen Einfluss. Die Dateien, die mit einem K beginnen, sind die sogenannten Kill-Skripte. Sie sind dafür verantwortlich, dass die Prozesse, die nicht zu diesem Runlevel gehören, gestoppt bzw. beim Verlassen des Runlevels beendet werden. Die Kill-Skripte werden auch oft Stop-Skripte genannt. Der `init`-Prozess ruft die Skripts mit dem Parameter `start` auf, wenn er ein S davor stehen hat, und mit dem Parameter `stop`, wenn er ein K davor stehen hat. Z. B. S88Web-Server `start`.

Ein genauerer Blick in das Verzeichnis (`ls -l`) zeigt uns, dass es sich nicht wirklich um Dateien in diesem Verzeichnis handelt, sondern vielmehr um einen Link auf Dateien im Verzeichnis `/etc/init.d`. Hier stehen die eigentlichen Skriptdateien. Diese Trennung ist sehr sinnvoll, da man die eigentlichen Start- und Stopp-Skripte nur in einem Verzeichnis warten muss und man damit auch einen besseren Überblick behält. Diese Dateien bzw. Skript-Dateien haben einen besonderen Aufbau:

```
#!/bin/bash
#Kommentare

case "$1" in
start)
echo "Einschalten des Prozesses"
;;
stop)
echo "Ausschalten des Prozesses"
;;
```

```

*)
    echo "Bitte geben sie als Parameter (start/stop)"
esac
exit 0

```

Wir erkennen sofort, dass dieses Skript so geschrieben ist, dass es sowohl den Prozess startet als auch stoppt. Wir überprüfen dies, indem wir in dem Verzeichnis `/etc/rc5.d/` die Start- und die Stop-Skripte daraufhin überprüfen, wohin die korrespondierenden S und K verlinkt sind.

Wenn alle Start-Skripte in diesem Verzeichnis, das den Runlevel definiert, ausgeführt wurden, befindet sich das System in diesem speziellen Modus. Dieser Modus kann mit dem `init`-Befehl gewechselt werden. Wenn wir den Befehl `#init 3` eingeben, wechselt das System vom Runlevel 5 in den Runlevel 3. Es werden alle Start-Skripte, die im Verzeichnis `/etc/rc3.d` stehen, ausgeführt. Das Programm **telinit** ist eigentlich nur auf das `init`-Programm gelinkt.

**Tabelle 93: Optionen für telinit**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
0,1,2,3,4,5,6	Wechselt in den angegebenen Runlevel.
a,b,c	Damit führt das <code>init</code> -Programm nur die Einträge aus der <code>inittab</code> -Datei aus, die die Runlevel a,b,c haben.
Q oder q	Die Datei <code>inittab</code> wird erneut ausgeführt.
S oder s	Geht zum Single-User-Mode über.

Eine Option, die das `telinit`-Programm noch zusätzlich besitzt, ist die Option `-t`, die dem `init`-Programm mitteilt, wie lange `init` warten soll zwischen den `SIGTERM`-Signal und dem `SIGKILL`-Signal. Der Standardwert ist 5 sec.

Die `init 0-6` versteht jede Linux-Maschine, die nach dem System-V-Startmechanismus arbeitet. So kann man einen Linux-

Rechner mit `init 0` herunterfahren und mit `init 6` einen Reboot einleiten. Zum Niederfahren einer Maschine kann man auch den Befehl ***shutdown*** verwenden.

```
#shutdown -h now
```

Dieser Befehl fährt das Linuxsystem sofort, also ohne Wartezeit, herunter.

**Tabelle 94: Optionen für shutdown**

<b><i>Optionen</i></b>	<b><i>Bedeutung</i></b>
-t	Sekunden zwischen Warnung und Kill-Signal.
-k	Schickt nur die Warnung an alle user, ohne den Rechner wirklich zu stoppen.
-r	Reboot.
-h	Halt.
-f	Übergeht den fsck beim Reboot.
-F	Erzwingt den fsck beim Reboot.
-c	Stoppt eine eingeleitete Abschaltung oder reboot.

Als Zeitargument kann man `now` für sofort verwenden oder `hh:mm` oder `hh`.

In der `inittab` weiter lesend, kommen wir zu den Einträgen:

```
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

Diese definieren die sogenannten virtuellen Terminals, die Linux unterstützt. Das ***mingetty***-Programm ist ein Terminal-Programm, in dem uns das Login-Programm empfängt. Die 6 virtuellen Konsolen oder Terminals können über die Tastenkombinationen

ALT und F1 bis F6 erreicht werden. Die Kombination ALT F7 bringt uns zur grafischen Oberfläche. Um von der grafischen Oberfläche auf die textbasierten virtuellen Terminals zu schalten, benötigen wir noch eine zusätzliche Taste, da die ALT-Taste unter den Windows-Managern eine besondere Bedeutung hat. Wir verwenden unter der grafischen Oberfläche die Tastenkombination ALT-STRG-F1 bis F6.

Wenn wir in den Runlevel 5, also in die grafische Oberfläche gebootet sind, wechseln wir mit ALT-STRG-F2 auf die zweite virtuelle Konsole.

Wir sehen eine Art Willkommensbotschaft z.B:

```
Red Hat Linux Release 9 (Shrinke)
Kernel 2.4.20-8 on an i686
```

Diese Ausgabe bzw. der anzuzeigende Text kann selbst konfiguriert werden. Der Text, der durch das `mingetty`-Programm angezeigt wird, ist in der Datei `/etc/issue` konfiguriert. Der Text, der in der Datei `/etc/issue` steht, wird so angezeigt, wie er eingegeben wurde. Allerdings gibt es eine große Anzahl von `@char` und `\char`-Zeichen, die vom `getty`-Programm unterstützt und ausgefüllt werden, die man auch verwenden kann. So ist z. B. die Kernelausgabe über `\r` und die Prozessorinformation über `\m` verfügbar. Unsere `/etc/issue` sieht also folgendermaßen aus:

```
Red Hat Linux Release 9 (Shrinke)
Kernel \r on an \m
```

So kann man für das `mingetty`-Programm folgende Sonderzeichen verwenden:

**Tabelle 95: Sonderzeichen**

<b>Zeichen</b>	<b>Bedeutung</b>
<code>\d</code>	Aktuelles Datum.
<code>\m</code>	Maschinenarchitektur.
<code>\n</code>	Netzwerkhosname.
<code>\o</code>	Domain Name.
<code>\r</code>	OS Release-Nummer.

\s	Operating System Name.
\u	Aktuelle Anzahl der User.
\v	OS-Versionsnummer.

Wenn wir über das Netzwerk mit dieser Maschine Kontakt z. B. mit telnet aufnehmen, bekommen wir einen ähnlichen Begrüßungsbildschirm (Banner). Die Meldung, die hier für Netzwerkbenutzer angezeigt wird, wird nicht aus der Datei `/etc/issue` ausgelesen, sondern aus der Datei **`/etc/issue.net`**. Für die `/etc/issue.net` gelten alle Fakten so wie für die `/etc/issue`.

#### *Anmerkung*

Die Datei `/etc/issue.net` sollte nicht zu freizügig mit Systeminformationen sein. Ein Angreifer kann sonst durch einen sogenannten **Portscan** und die daraus resultierende **Banneranalyse** zu leicht auf Betriebssystemart und Version und, und, und schließen. Auf keinen Fall darf Willkommen oben stehen, denn dies ist eine Einladung. Am sinnvollsten erscheint mir, einen Text anzuzeigen, der besagt, dass dieses System geschützt und nur für privilegierte Benutzer vorgesehen ist und deshalb alle Einbruchversuche protokolliert und zur Anzeige gebracht werden.

Nach dem Anmelden an unserem System bekommen wir einen weiteren Begrüßungsbildschirm. Diesen können wir auch etwas beeinflussen, indem wir die Datei **`/etc/motd`** (Motto des Tages) editieren. Der Text aus dieser Datei wird ausgelesen und beim Begrüßungsbildschirm mit angezeigt. Wenn man diese Datei automatisiert mit neuen Inhalten füllen lässt, bietet dies einen besonderen Reiz. Man kann diese Datei auch als Infodatei für die Benutzer verwenden.

## 6.3

### **Big Brother – Alles wird protokolliert**

Bei einem so komplexen mehrbenutzer-, multitasking-System, wie es Linux darstellt, spielt die Protokollierung von verschiedenen Services und Vorgängen eine zentrale Rolle. Die Überwachung und Analyse von den verschiedenen Logfiles bzw. Protokolldateien ist eine der zentralen Aufgaben eines Systemadmi-

nistrators. Um so wichtiger sind die Themen Shell-Programmierung und Automatisierung für die tägliche Routinearbeit. Wir wollen uns mit dem Systemprotokoll und dem Dienst, der diese Protokolle verwaltet und schreibt, beschäftigen. Der **syslogd** ist jener Serverdienst, der uns hier zu Diensten sein wird.

### Einschub

Wir sind endlich bei der dritten Art von Prozessen angelangt. Wir kennen bereits Vordergrundprozesse und Hintergrundprozesse. Diese Prozesse sind jeweils an ein aufrufendes Terminal gebunden und haben einen Standard-Ausgabe- und einen Standard-Eingabe- sowie einen Standard-Fehler-Kanal. Die nun dazu kommende Gruppe von Prozessen, zu denen auch der syslogd gehört, heißen **Daemon**-Prozesse. Daemon deswegen, weil diese Prozesse die Eigenschaft haben, nicht an ein aufrufendes Terminal gebunden zu sein und keinen Standard-Eingabe-, Standard-Ausgabe- und Standard-Fehler-Kanal zu besitzen. Diese Prozesse sind die eigentlichen Diener oder besser Dienste, also Serverdienste, die auf unserem System laufen. Als Namenskonvention für diese Serverdienste, Daemons, hat sich eingebürgert, dass man ein kleines d für Daemon hinter den eigentlichen Namen hinzufügt.

Der syslogd (gesprochen syslog-Daemon) ist also unser Protokollschreiber. Dieser Dienst wird über eine Konfigurationsdatei **/etc/syslog.conf** konfiguriert. Gestartet wird er über den **init**-Prozess. Wenn wir auf unserem installierten System (Red Hat 9) einen Blick in das Verzeichnis **/etc/rc5.d** machen, dann finden wir dort einen Link mit dem Namen **S12syslog**. Der syslogd kann mit verschiedenen Optionen aufgerufen werden.

### Aufgabe

Versuchen Sie in der Datei **S12syslog** herauszufinden, mit welchen Optionen der syslogd gestartet wird.

**Tabelle 96: Optionen für den syslog Daemon**

<b>Optionen</b>	<b>Bedeutung</b>
-a	Damit kann man die Sockets definieren, über die der Daemon kommuniziert.
-d	Schaltet den debugging-Mechanismus ein.
-f FILE	FILE wird anstelle der Datei <b>/etc/syslog.conf</b> verwendet.

-h	Damit wird der syslogd angewiesen, an konfigurierte Computer syslog-Meldungen weiterzuleiten, die über das Netzwerk erhalten wurden.
-l	Liste von zu loggenden Maschinen. Die Angabe erfolgt mit reinen Hostnamen und wird mit : getrennt.
-m	Intervall, an dem ein Zeitstempel markiert wird. Default ist 20 min.
-r	Erlaubt es dem syslogd, Meldungen über das Netzwerk zu empfangen.
-x	Deaktiviert die Namensauflösung, wenn -r aktiviert ist.

Sehen wir uns das Konfigurationsfile für den syslogd an. Die Datei ist in Felder aufgeteilt. Das erste Feld (ursprung.level) bestimmt, welche Meldungen aufgezeichnet werden, und das zweite Feld gibt den Ort an, wohin diese Meldungen geschrieben werden sollen.

Der Ursprung gibt die Systemkomponente oder das Programm an, von dem die Meldungen kommen, und der Level gibt die Gewichtung der Meldung wieder. In abnehmender Gewichtung der Meldung können folgende Level unterschieden werden: emerg, crit, err, warning, notice, info, debug.

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.* /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none /var/log/messages

# The authpriv file has restricted access.
authpriv.* /var/log/secure

# Log all the mail messages in one place.
mail.* /var/log/maillog
```



```
# Log cron stuff
cron.*                /var/log/cron

# Save boot messages also to boot.log
local7.*              /var/log/boot.log
```

Wir sehen darin Einträge der Form:

```
*.info;mail.none;authpriv.none;cron.none    /var/log/messages
```

Das bedeutet, dass egal von welcher Komponente des Systems, also \*, die Meldung in der angegebenen Gewichtung kommt, sie wird in die Datei **/var/log/messages** geschrieben. Wir erkennen auch, dass das Verzeichnis **/var/log/** anscheinend das Protokollverzeichnis darstellt, denn auch andere Meldungen werden in dieses Verzeichnis, allerdings mitunter in eine andere Datei/Unterverzeichnis, geschrieben.

Die unumstrittene Königin aller Logdateien (Protokolldateien) ist aber die Datei **/var/log/messages**. Dort landet wirklich fast alles. Diese Datei ist für den Systemadministrator auch die erste Anlaufstelle, wenn irgendetwas am System nicht so läuft wie es soll. In der Datei **/var/log/messages** findet man wahrscheinlich mehr oder weniger hilfreiche Angaben über die Art der Schiefelage. Da wir in dieser Datei sehr viel protokollieren, kann sie auf einem virulenten System sehr schnell sehr stark anwachsen. Wir müssen uns somit um ein System kümmern, welches die Logdateien und deren Größe im System verwaltet.

Auszug aus der Datei **/var/log/messages**:

```
Sep 3 10:39:07 tux kernel: Adding Swap: 257032k swap-space (priority -1)
Sep 3 10:39:07 tux autofs: automount startup succeeded
Sep 3 10:39:07 tux kernel: kjournald starting. Commit interval 5 seconds
Sep 3 10:39:07 tux kernel: EXT3-fs: mounted filesystem with ordered data mode.
Sep 3 10:39:12 tux sendmail: Starten von sendmail succeeded
Sep 3 10:39:12 tux sendmail: Starten von sm-client succeeded
Sep 3 10:39:12 tux gpm: Starten von gpm succeeded
Sep 3 10:39:13 tux crond: Starten von crond succeeded
Sep 3 10:39:15 tux kernel: parport0: PC-style at 0x378 (0x778) [PCSP,TRISTATE]
Sep 3 10:39:15 tux kernel: parport0: irq 7 detected
Sep 3 10:39:15 tux kernel: lp0: using parport0 (polling).
Sep 3 10:39:15 tux kernel: lp0: console ready
Sep 3 10:39:15 tux modprobe: modprobe: Can't locate module char-major-188
Sep 3 10:39:15 tux last message repeated 15 times
Sep 3 10:39:15 tux cups: Starten von cupsd succeeded
```

```
Sep  3 10:39:16 tux xfs: Starten von xfs succeeded
Sep  3 10:39:16 tux anacron: Starten von anacron succeeded
Sep  3 10:39:16 tux atd: Starten von atd succeeded
Sep  3 10:39:36 tux gdm(pam_unix)[1837]: session opened for user root by (uid=0)
Sep  3 15:02:29 tux login(pam_unix)[1790]: session opened for user root by
LOGIN(uid=0)
Sep  3 15:02:29 tux  -- root[1790]: ROOT LOGIN ON tty3
Sep  3 15:02:31 tux login(pam_unix)[1790]: session closed for user root
Sep  3 15:24:12 tux login(pam_unix)[1789]: session opened for user root by
LOGIN(uid=0)
Sep  3 15:24:12 tux  -- root[1789]: ROOT LOGIN ON tty2
Sep  3 16:39:19 tux kernel: floppy0: obsolete eject ioctl
```

### *Hinweis*

Zum Beobachten der Datei `/var/log/messages` bietet sich der Befehl `tail` mit der Option `-f` an. Damit kann man die Datei offen halten und alle Änderungen der Datei, also was hinten angefügt wird, live miterleben. Man ist quasi in Echtzeit dabei.

Damit unsere Logdateien nicht unendlich anwachsen und Platz verschwenden, sollten sie in regelmäßigen Abständen archiviert und gelöscht werden. Das Tool, das uns dabei behilflich sein kann, nennt sich **logrotate**. Das `logrotate`-Programm rotiert, aber nicht im wörtlichen Sinn, sondern vertauscht und komprimiert Systemlogdateien. Der Befehl `logrotate` sollte nicht im Konsolenfenster manuell aufgerufen werden, sondern automatisiert in einem täglichen, wöchentlichen oder monatlichen Rhythmus ablaufen. Deshalb werden wir hier nur auf die möglichen Optionen eingehen.

**Tabelle 97: Optionen für logrotate**

<b>Optionen</b>	<b>Bedeutung</b>
<code>-d</code>	Schaltet den debugging-Modus ein.
<code>-f</code>	Weist <code>logrotate</code> an, sich auszuführen, auch wenn es glaubt, nichts machen zu müssen.
<code>-m command</code>	Definiert das Kommando, mit welchem <code>logrotate</code> mails verschickt. Der Standard ist <code>/bin/mail -s</code> .

Das Programm `logrotate` wird durch eine Konfigurationsdatei **`logrotate.conf`**, die wiederum im `/etc`-Verzeichnis gefunden werden kann, konfiguriert. In diese Konfigurationsdatei wird der Inhalt des Verzeichnisses **`/etc/logrotate.d`** eingelesen und integriert.

```
# sample logrotate configuration file
compress

/var/log/messages {
    rotate 5
    weekly
    postrotate
        /sbin/killall -HUP syslog
    endscript
}

"/var/log/httpd/access.log"

/var/log/httpd/error.log {
    rotate 5
    mail www@my.org
    size=100k
    postrotate
        /sbin/killall -HUP httpd
    endscript
}
```

Die wichtigsten Parameter, die in der Datei `logrotate.conf` auftreten können:

**Tabelle 98: logrotate-Parameter in der Konfigurationsdatei**

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
<code>compress</code>	Alte Versionen werden mit dem Tool <code>gzip</code> komprimiert.
<code>copy</code>	Macht eine Kopie des Logfiles und ändert das Original nicht.

daily	Logfiles werden jeden Tag rotiert.
mail adresse	Das rotierte Log wird an die adresse gemailt.
missingok	Wenn ein Logfile nicht gefunden werden kann, gehe zum nächsten Punkt ohne Fehlermeldung.
monthly	Monatliches Rotieren.
postrotate/endscript	Alles was zwischen diesen beiden Parametern steht, wird nach dem Rotieren ausgeführt.
rotate Nummer	Die Logdateien werden Nummer Mal rotiert, bevor Sie gelöscht werden.
size N	Logdateien werden rotiert, wenn sie größer als N (Bytes) werden. Die Zusatzparameter für kB und M für MB sind möglich.
weekly	Wöchentliches Rotieren.

Damit die Protokolle und das System insgesamt Sinn machen und auch aussagekräftig sind, muss auf jeden Fall die Systemzeit stimmen, damit die Einträge in den Logdateien mit dem aktuellen Zeitstempel versehen werden können.

Mit dem Befehl **date** können wir die aktuelle Systemzeit abfragen und setzen.

```
# date
Don Sep    4 09:02:15 CEST 2003
```

**Tabelle 99: Optionen für date**

<b>Optionen</b>	<b>Bedeutung</b>
-s STRING	Setzt die Zeit auf STRING und ist in der Form MDDhhmm[[CC]YY][.ss].
-u	Setzt oder zeigt die Zeit in Universal Time an.
-ITIMESPEC	Gibt die Zeit im ISO 8601-Format aus.

An die Befehlszeile können noch Formatangaben angehängt werden, die bestimmen, in welcher Form das Datum ausgegeben wird.

```
#date +%a
```

```
Don
```

**Tabelle 100: Formatangaben bei date**

<b>Formatsring</b>	<b>Bedeutung</b>
%a	Wochentag in Kurzschreibweise.
%A	Wochentag in Langformat.
%b	Monat in Kurzform.
%B	Monat in Langform.
%C	Jahrzehnt.
%d	Tag der Woche.
%D	Datum im Format mm/dd/yy.
%e	Tag des Monats.
%H	Stunde im 24h-Format.
%I	Stunde im 12h-Format.
%j	Tag des Jahres.
%M	Minuten.

In einer internationalen Umgebung ist das Setzen der Zeit auf eine UTC-Form sinnvoll.

Bei der Installation sind wir nach der Zeitzone, in der wir leben, gefragt worden. Diese Zeitzonen sind in dem Verzeichnis **/usr/share/zoneinfo** zusammengestellt. In diesem Verzeichnis sind alle Zoneninformationen in Binärdateien gesammelt und eventuell in Unterverzeichnisse aufgeteilt. So ist z. B. ein Verzeichnis Europe zu finden, in denen die europäischen Hauptstädte aufgelistet sind. Wenn wir uns wieder an den Installationsprozess erinnern, wissen wir, dass wir auch gefragt worden sind, ob unser Computer auf UTC oder Lokalzeit eingestellt werden soll. Wir haben die Option Lokalzeit gewählt. Damit diese Einstellung mit unserer Zeitzone übereinstimmt, wird nichts anderes gemacht, als ein Link in das Verzeichnis /etc. Dieser Link heißt **localtime** und zeigt auf die gewünschte Zeitzone. In meinem Fall die von Wien (Vienna).

```
#ls -l /etc/localtime
lrwxrwxrwx 1 root root 33  4. Sep 09:24 /etc/localtime
-> /usr/share/zoneinfo/Europe/Vienna
```

Wenn wir unsere Zeitzone ändern wollen, müssen wir einfach den Link von localtime auf eine andere, die gewünschte Zeitzone setzen.

Wenn bei einem einzelnen Computer die Zeit schon eine sehr wichtige Rolle spielt, ist die korrekte Zeit in einem Netzwerk unumgänglich. Wenn unterschiedliche Rechner im Netzwerk unterschiedliche Zeiten haben, weiß man nie, wann man von wem ein Dokument bekommen hat und welches das Aktuelle ist. Abhilfe schafft hier ein Zeitserver, der **ntpd** (Network Time Protocol Daemon), der über die Start-Skripte gestartet werden kann, soll und muss.

Man kann mit einer offenen Internetverbindung auch einen der vielen im Netz vorhandenen Timeserver verwenden. Linux liefert uns mit dem ntp-Daemon aber schon alles Notwendige mit, um einen eigenen Zeitserver zu unterhalten.

Wenn dieser Zeitserver gestartet wird, wird zuerst die Datei /etc/ntp.conf eingelesen, damit der Zeitserver weiß, wie er sich verhalten soll. Als Server, der selbst die Zeit weitergibt oder als Dienst, der im Hintergrund einen anderen Zeitserver befragt und die Systemzeit stellt.

In der Datei /etc/ntp.conf steht dann z. B. folgendes:

```
# ntp.conf Zeitsdrvrkonfig
server ts1.univie.ac.at
server ts2.univie.ac.at

driftfile /etc/ntp/ntp.drift
enable monitor
restrict default notrust nomodify
restrict 10.0.1.2 nomodify
restrict 120.0.0.1
```

Diese Konfigurationsdatei tut Folgendes: Der Hauptserver, von dem unser System die aktuelle Uhrzeit bezieht, ist ts1.univie.ac.at, als Ersatzserver wird der ts2.univie.ac.at angegeben. Die ermittelte Drift des Systems gegenüber dem Zeitserver wird in der Datei **driftfile** gespeichert, damit sie nicht jedesmal neu ermittelt werden muss. Dies gewährleistet der aktivierte Monitor. Damit keine Rechner, außer den unten genannten, auf den Zeitserverdienst und dessen Konfiguration zugreifen können, kommen die **restrict**-Einträge zum Tragen. Damit wird konfiguriert, dass nur der Rechner 10.0.1.2 auf den Dienst zugreifen kann, aber die Konfiguration nicht ändern darf. Dieses Recht hat nur der localhost 127.0.0.1.

Bei einem anderen Linux-Rechner, der als Client arbeiten soll, braucht in der ntp.conf nur der Server im eigenen Netzwerk angegeben zu werden – voilà.

Wenn man die Systemzeit über einen Server manuell einstellen möchte, verwendet man dazu den Befehl **ntpdate**.

Mit ntpdate und Servername kann man die Systemzeit einstellen.

```
#ntpdate servername
```

Zum Abschluss kommen wir noch auf die sogenannte **Real Time Clock** (RTC) zu sprechen. Das ist die Uhr auf dem Motherboard – also die Hardware-Uhr.

Mit dem Befehl **hwclock** kann man die RTC abfragen und setzen.

```
# hwclock --show
```

```
Don 04 Sep 2003 11:07:49 CEST -0.974248 Sekunden
```

**Tabelle 101: Optionen für hwclock**

<b>Optionen</b>	<b>Bedeutung</b>
--show	Liest die RTC aus.
--set	Setzt die RTC auf die Zeit, die durch date gegeben ist.
--hctosys	Setzt die Systemzeit nach der RTC.
--systohc	Setzt die RTC nach der aktuellen Systemzeit.
--adjust	Addiert und subtrahiert die Drift.
--date=STRING	Setzt die RTC auf STRING („9/22/99 16:45:05“).
--utc	Gibt Zeit in UTC aus.

Bevor wir uns unter dem Motto „Alles neu macht der Mai“ mit dem Hinzufügen von neuer Hardware und Peripheriegeräten beschäftigen, sehen wir uns das Software-Paket-Managemet und den Kernel mit seiner Konfiguration etwas genauer an. Beim Hinzufügen von neuer Software bzw. Hardware ist es unter Umständen notwendig, einen neuen Kernel „bauen“ zu müssen, da die Hardwareunterstützung im Kernel liegt.



Der Kernel ist die zentrale Komponente und das eigentliche Betriebssystem. Die unzähligen Tools und Programme, die sich rund um den Kernel scharen, sind im Grunde nur Zugaben. Wenn man von Linux spricht, spricht man eigentlich nur vom Kernel. So sollte man, wenn man vergleicht, besser von Linux-Distributionen sprechen. Natürlich hat jeder Distributor auch Veränderungen am Kernel vorgenommen, so dass er an seine Zielgruppe und deren Bedürfnisse optimal angepasst ist.

Die Hauptaufgaben des Kernels sind die Hardwareunterstützung und die Verwaltung der Prozesszeit und des zugeteilten Speichers. Er trennt Hardware und Anwendungsprogramme.

Gründe, warum man einen neuen Kernel „bauen“ soll oder muss, sind hauptsächlich dadurch gegeben, dass man neue Hardware installiert, die vom derzeitigen Kernel nicht oder nur unzureichend unterstützt wird bzw. wenn das System nicht stabil läuft und die neuere Kernel-Version dafür Abhilfe verspricht. Oft spielen aber auch Performancegründe eine Rolle, da eventuell der neue Kernel modernere Prozessmanagementsystematiken besitzt.

Es hat sich immer mehr durchgesetzt, dass viele Hardwareunterstützungen nicht im eigentlichen Kernel fix eingebunden werden, sondern als sogenannte Kernel-Module bei Bedarf dynamisch in den Kernel „eingehängt“, also hinzugefügt und auch wieder aus dem Kernel herausgenommen werden können. Diese Modultechnik erlaubt es, den eigentlichen Kernel klein zu halten und somit mehr Flexibilität zu erhalten, da man nur die jeweiligen Module übersetzen muss und sie in den laufenden Kernel einbinden kann. So können Treiber unabhängig vom Kernel entwickelt werden.

### 7.1

## Das Herz des Systems

Einen neuen Kernel bauen heißt, ihn aus seinem Quelltext, also dem Sourcecode, neu zu übersetzen (compilieren). Welcher Compiler (Übersetzer) und welche Bibliotheken man verwenden

soll, ist in einer **README**-Datei bei den Quellen hinterlegt. Das heißt aber, wir brauchen die Quellen für den Kernel. Die meisten Linux-Distributionen liefern die von ihnen verwendeten Kernel-Sourcen auch auf CD oder DVD mit aus. Man kann also die Sourcen, sofern sie nicht ohnehin schon mitinstalliert sind, von der CD oder DVD nachinstallieren. Diese Kernel-Versionen sind in den meisten Fällen allerdings nicht die aktuellsten. Will man den aktuellsten Kernel für seine Zwecke verwenden, bleibt oft nichts anderes übrig als die Kernel-Quellen von der Web-Site **[www.kernel.org](http://www.kernel.org)** herunter zu laden und in das richtige Verzeichnis zu installieren.

Das Verzeichnis, in dem die Sourcen zu finden bzw. zu installieren sind, ist **/usr/src/**. In diesem Verzeichnis findet man normalerweise Verzeichnisse wie **linux** oder **linux-2.4.**, wobei die Zahl 2.4 die Versionsnummer des Kernel angibt. In diese Verzeichnisse sollten die Quellen installiert werden.

Wir wollen nun einen neuen Kernel bauen. Wir machen dies nur so aus Spaß, da wir eigentlich keinen triftigen Grund dazu haben. Wir holen uns einen aktuellen Kernel aus dem Internet. Dies ist in unserem Fall der Kernel mit der Versionsnummer **2.5.9**, und die Datei, die wir in unserem root-HOME-Verzeichnis speichern, heißt **linux-2.5.9.tar.gz** und ist ca. 32MB groß.

#### *Einschub*

Die Endungen **gz** und **tar** sagen uns, dass es sich dabei um eine komprimierte Archivdatei handelt. Um mit diesem Archiv umgehen zu können, benötigen wir noch die Befehle **tar** und **gzip**. Ich möchte in einem kleinen Einschub diese Befehle kurz darstellen.

Der Befehl **gzip** steht für Gnuzip. Wir kennen zip-Dateien von Windows her, wo man oft mit dem Winzip-Programm diese komprimierten Dateien erstellt und auch wieder dekomprimiert.

```
# gzip -d linux-2.5.9.tar.gz
```

Mit diesem Befehl können wir das komprimierte Archiv dekomprimieren.

**Tabelle 102: Optionen für gzip**

<b>Optionen</b>	<b>Bedeutung</b>
-c	Schreibt auf die Standard-Ausgabe und lässt die eigentlichen Dateien unverändert.
-d	Dekompress.
-f	Erzwingt das Komprimieren und Dekomprimieren unabhängig davon, ob Dateien bereits existieren.
-l	Listet die Daten für komprimierte Größe, unkomprimierte Größe, Kompressionsrate und unkomprimierten Namen auf.
-n	Bei der Kompression wird nicht der Originalname und Zeitstempel gespeichert.
-N	Standard. Bei der Kompression wird der originale Dateiname und Zeitstempel verwendet.
-q	Quiet. Es werden keine Warnungen ausgegeben.
-r	Rekursive Ausführung (für Directories).
-t	Überprüft die Kompressionsintegrität.
-#	# ist eine Zahl von 1 (schnell) bis 9 (beste), die die Kompression angibt.

Der gzip-Befehl erwartet von einer komprimierten Datei die Endung gz. Er hängt diese Endung auch automatisch an die Datei an.

Nun fehlt nur noch der tar-Befehl. Der Name tar leitet sich von tape archiv ab. Das sagt eigentlich schon die ganze Bedeutung

und Aufgabe des tar-Befehls aus. Die grundlegende Aufgabe von tar ist es, Daten auf einem Bandlaufwerk zu archivieren. Der tar-Befehl ist ein extrem mächtiger. Wir können nicht nur ein Archiv auf einem Bandlaufwerk anlegen und einlesen, sondern auch auf einer Festplatte. So können wir z. B. unseren Kernel, der nach dem dekomprimieren linux-2.5.9.tar heißt, mit dem tar-Befehl weiter bearbeiten.

Zuerst wollen wir nachsehen, was in diesem Archiv überhaupt enthalten ist.

```
# tar -tvf linux-2.5.9.tar | more
drwxrwxr-x torvalds/users    0 2002-04-23 00:29:26 linux-2.5.9/
drwxrwxr-x torvalds/users    0 2002-04-23 00:29:54 linux-2.5.9/fs/
drwxrwxr-x torvalds/users    0 2002-04-23 00:29:44 linux-2.5.9/fs/bfs/
-rw-r--r-- torvalds/users  100 2002-04-23 00:27:41 linux-
2.5.9/fs/bfs/bfs_defs.h
-rw-r--r-- torvalds/users   398 2002-04-23 00:29:41 linux-
2.5.9/fs/bfs/Makefile
```

.....

Wir erkennen, dass alles in einem Verzeichnis mit dem Namen linux-2.5.9 zusammengefasst ist. Wir wollen dieses Verzeichnis in unserm Pfad /usr/src entpacken. Dazu kopieren wir die Datei linux-2.5.9.tar in das Verzeichnis /usr/src. Dort verwenden wir den Befehl:

```
# tar -xvf linux-2.5.9.tar
linux-2.5.9/
linux-2.5.9/fs/
linux-2.5.9/fs/bfs/
...
```

Das Archiv ist damit entpackt, und wir finden das Verzeichnis /usr/src/linux-2.5.9 vor, in dem alle Kernel-Sourcecode-Dateien enthalten sind. Bevor wir mit dem Kernelbauen weiter vorschreiten, stellen wir noch die wichtigsten Optionen für den tar-Befehl zusammen.

**Tabelle 103: Optionen für tar**

<b>Optionen</b>	<b>Bedeutung</b>
-C	Erzeugt ein Archiv.
-X	Entpackt ein Archiv.

-t	Listet den Archivinhalt auf.
-f DATEI	Stellt DATEI als Ein-Ausgabekanal anstelle des Default-Bandlaufwerkes /dev/st0 ein.

Mit dem tar-Befehl können sehr elegant, komplexe Backupstrukturen aufgebaut werden.

Wir wissen, dass wir einen Kernel mit der Versionsnummer 2.5.9 installieren wollen. Da wir im Verzeichnis /usr/src ein Verzeichnis linux-2.4 gefunden haben, nehmen wir an, dass auf unserer Maschine auch ein Kernel mit der Versionsnummer 2.4 läuft.

Um dies zu überprüfen, verwenden wir das Tool **uname**.

```
# uname -r
2.4.20-8
```

**Tabelle 104: Optionen für uname**

<b>Optionen</b>	<b>Bedeutung</b>
-a	Gibt alle Informationen aus.
-s	Kernel-Name.
-n	Hostnamen.
-r	Kernel-Release.
-v	Kernel-Version.
-m	Hardwarearchitektur.
-p	Prozessor.
-i	Hardware-Plattform.
-o	Betriebssystem.

Damit wissen wir mit Sicherheit, dass unsere Kernel-Sourcen neuer sind als der aktuell laufende Kernel. Was sagen aber die Nummern 2.5.9 bzw 2.4.20-8 aus? Die Zahl 2 gibt die Kernel-Version, die 5 den Patchlevel und die 9 den Sublevel an. Bei Änderungen kommt hier noch zum Sublevel eine Extraversion hinzu wie bei unserm 2.4.20-8. Das heißt, Version 2 Patchlevel 4 und Sublevel 20 mit Extralevel 8.

Nun müssen wir den neuen Kernel konfigurieren. Dies kann über den Befehl **make** mit einem sogenannten **Target** geschehen. Für die Konfiguration stehen uns folgende Targets zur Verfügung. Das **config**-Target, welches eine Kette von Fragen bildet und nicht sehr komfortabel ist. Das **menuconfig**-Target, welches uns ein textbasiertes Menü für die Konfiguration anbietet. Das **xconfig**-Target ist sicherlich das komfortabelste, benötigt aber, wie der Name schon anklingen lässt, das X11 Fenster-System.

Die meisten in diesen Menüs auswählbaren Fragen können mit Ja (y) oder Nein (n) beantwortet werden. Viele haben aber auch die Möglichkeit vorgesehen, dass man diese Option oder Eigenschaft als Modul (m) in den Kernel integriert. Zu den meisten Fragen gibt es auch einen kleinen Hilfetext, der aber meistens nicht sehr aussagekräftig ist.

Wir verwenden das textbasierte Menü.

```
# make menuconfig
```

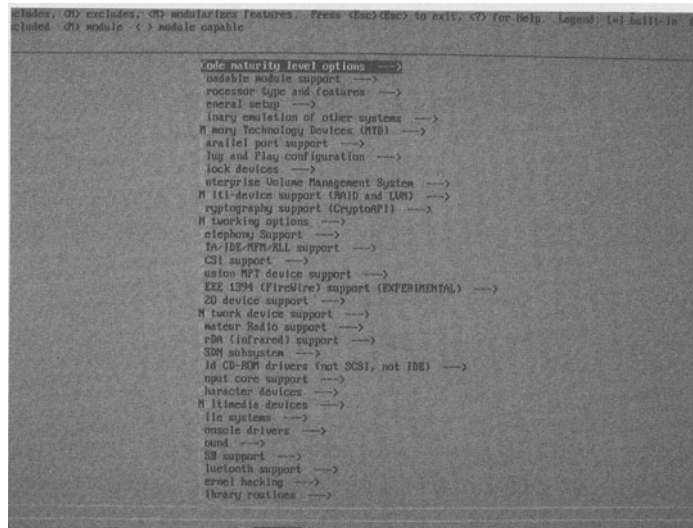


Abbildung 10: Menuconfig

Wir befinden uns im Hauptmenü. Die einzelnen Menüpunkte sind über die Pfeiltasten zu erreichen und werden mit der Eingabe-Taste selektiert.

Wir wollen einen Kernel erstellen, der mit einer NTFS-Partition umgehen kann. Dazu finden wir im Hauptmenü einen Menüpunkt der „**File systems**“ heißt. Wir steuern diesen Menüpunkt mit den Pfeiltasten an und selektieren ihn mit Return.

```

show, ON excludes, ON modularizes features, (F)resh (R)estore to exit, (Y) for help, Legend: (w) with-on
cluded (M) module (<*) module capable

(O) JFS debugging verbosity (0 = quiet, 3 = noisy)
(I*) JFS stats available in /proc/filesystems
(O) Journalling Flash File System v2 (JFFS2) support
(O) JFFS2 debugging verbosity (0 = quiet, 2 = noisy)
(O) compressed ROM file system support
(I*) virtual memory file system support (former shs fs)
(<*) 30 9660 CDROM file system support
(I*) M cross-ol Joliet CDROM extensions
(I*) transparent decompression extension
(O) FS filesystem support
(I*) FS Posix ACL support
(I) FS debugging
(I*) FS statistics
(<*) Minix fs support
(O) reeVFS file system support (VERITAS VxFS(TM) compatible)
(<*) NTFS file system support (read only)
(I) M FS debugging support
(I) M FS write support (DANGEROUS)
(O) 3-2 NIFS file system support
(I*) /proc file system support
(I*) /proc/config.gz (kernel configuration)
(I) /ev file system support (EXPERIMENTAL)
(I*) /ev/pts file system for Unix98 PTys
(O) MM4 file system support (read only)
(I) MM4FS write support (DANGEROUS)
(O) OM file system support
(<*) econd extended fs support
(I*) xt2 extended attributes
(I*) xt2 extended attribute block sharing
(I*) xt2 extended user attributes
(I*) xt2 extended extended attributes
(I*) xt2 POSIX Access Control Lists
(O) system V/Xenix/V7/Coherent file system support
(O) BF file system support (read only)

```

Abbildung 11: Menuconfig - Filesystem

In dem folgenden Menü suchen wir den Eintrag „**NTFS file system support (read only)**“ und aktivieren ihn mit der Leertaste. Wir erkennen, dass in den eckigen Klammern ein **<M>** erscheint, das uns angibt, dass diese Option als Modul eingebunden werden soll. Wenn wir die Leertaste ein weiteres Mal drücken, ändert sich der Zeilenbeginn in **<\*>**, was uns anzeigt, dass diese Option fest in den Kernel integriert wird. Wir belassen es aber beim Modul.

Mit den Pfeiltasten bewegen wir den Cursor auf Exit und bestätigen mit der Eingabe-Taste. Wieder im Hauptmenü angelangt, gehen wir auf Exit und erhalten die Abfrage, ob wir die Änderungen in der Konfigurationsdatei speichern wollen oder nicht. Wir bestätigen dies mit Yes. Die Datei, in der die Konfiguration gespeichert wird, heißt standardmäßig **/usr/src/linux-2.5.9/config**.

Nun wollen wir die Abhängigkeiten unseres Kernels ermitteln und diese konfigurieren lassen. Wir verwenden dazu:

```
#make dep
```

Nun kommt die eigentliche Knochenarbeit für unseren Rechner, den Kernel kompilieren.

```
#make zImage
```

Nach der Kompilation liegt der neu kompilierte Kernel im Verzeichnis ***/arch/i386/boot*** und heißt ***zImage***. Wenn man sehr viele Optionen des Kernels direkt in den Kernel integriert, kann es vorkommen, dass das Kernel-Image zu groß wird. In diesem Fall bricht der Kompilierungsprozess mit einer entsprechenden Fehlermeldung ab. Wir können darauf reagieren, indem wir ***bzImage*** als Target beim make-Befehl angeben. Bei unserem Kernel ist genau dies passiert, und wir starten alles neu mit dem Target bzImage.

```
#make modules
```

Dieser Befehl übersetzt die als Modul gekennzeichneten Teile des Kernels. Mit dem Befehl

```
#make modules_install
```

werden die Kernel-Module in das Verzeichnis ***/lib/modules/2.5.9*** installiert.

Mit dem Befehl `make install` kann der neue Kernel zum aktuellen geschrieben werden.

Der hier beschriebene Vorgang führt zu einem nicht funktionsfähigen Kernel, da wir auf unsere Hardware und gewünschten Optionen näher eingehen müssten. Dies würde aber weit über den Rahmen dieses Buches hinausgehen.

Wenden wir uns lieber der Verwaltung von dynamisch ladbaren Modulen unseres Kernels zu. Um herauszufinden, welche Module gerade in meinen Kernel eingebunden sind, verwenden wir den Befehl ***lsmod*** (list modules).

```
#lsmod
```

Module	Size	Used by	Not tainted
usb-uhci	26348	0	(unused)
nls_iso8859-1	3516	1	(autoclean)
nls_cp437	5116	1	(autoclean)



vfat	13004	1	(autoclean)
fat	38808	0	(autoclean) [vfat]
parport_pc	19076	1	(autoclean)
lp	8996	0	(autoclean)
parport	37056	1	(autoclean) [par-
port_pc lp]			
autofs	13268	0	(autoclean) (unused)
8139too	18088	1	
mii	3976	0	[8139too]
sg	36524	0	(autoclean)
sr_mod	18136	0	(autoclean)
ide-scsi	12208	0	
scsi_mod	107160	3	[sg sr_mod ide-scsi]
ide-cd	35708	0	
cdrom	33728	0	[sr_mod ide-cd]
keybdev	2944	0	(unused)
mousedev	5492	1	
hid	22148	0	(unused)
input	5856	0	[keybdev mousedev hid]
usbcore	78784	1	[usb-uhci hid]
ext3	70784	2	
jbd	51892	2	[ext3]

Wir finden ein Modul **usb-ubsci**, welches für die USB-Unterstützung zuständig ist. Nun wollen wir dieses Modul aus dem Kernel wieder entfernen. Dazu verwenden wir den Befehl **rmmod** (remove module).

**Tabelle 105: Optionen für rmmod**

<b>Option</b>	<b>Bedeutung</b>
-a	Autoclean. Markiert unbenötigte Module als bereinigt und entfernt bereits markierte Module.
-e	Speichert persistente Daten für das angegebene Modul, ohne es zu entfernen.
-s	Gibt alles zum syslogd aus anstelle am Terminal.

-r	Entfernt einen Modulstack.
-v	Verbose.

```
#rmmod usb-uhci
```

Wenn wir `lsmod` erneut aufrufen, erkennen wir, dass das Modul `usb-uhci` nicht mehr aufgelistet wird.

Wir wollen dieses Modul doch wieder in den Kernel einhängen, da wir ein USB-Gerät bekommen haben und es auch anschließen wollen. Zum Einhängen von Modulen in den Kernel benutzen wir den Befehl ***insmod*** (insert module).

**Tabelle 106: Optionen für *insmod***

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
-e Name	Name spezifiziert, wohin und von wo persistente Daten eines Moduls gelesen bzw. geschrieben werden sollen.
-f	Erzwingt das Laden eines Moduls, wenn die Versionsnummer des laufenden Kernels nicht mit der Nummer übereinstimmt, für die das Modul kompiliert wurde.
-k	Setzt das autoclean-Flag auf das Modul.
-L	Verhindert das gleichzeitige mehrfache Laden des Moduls .
-m	Gibt auf der Standard-Ausgabe Informationen aus, die beim Beheben von Fehlern hilfreich sein können.
-n	Probelauf - lädt aber das Modul nicht.
-p	Probe, ob das Modul erfolgreich geladen werden kann.

-q	Gibt keine Meldungen aus.
-r	Notwendig, wenn Module nicht als root kompiliert wurden.
-s	Schickt alles zum syslogd anstelle des Terminals

Wenn wir unser Modul wieder einhängen wollen, verwenden wir den folgenden Befehl:

```
#insmod usb-uhci
#
```

Um zu überprüfen, ob das Modul tatsächlich wieder im Kernel integriert ist, benutzen wir wieder `lsmod`.

Betrachten wir das Modul `8139too`. Wir wollen wissen, welches Modul es ist und was es tut. Dazu benutzen wir den Befehl ***modinfo*** (module info).

```
#modinfo 8139too
filename:      /lib/modules/2.4.20-8/kernel/drivers/net/8139too.o
description:   "RealTek RTL-8139 Fast Ethernet driver"
author:        "Jeff Garzik <jgarzik@pobox.com>"
license:       "GPL"
parm:          multicast_filter_limit int, description "8139too maximum
...
```

Wir erkennen daraus, dass es sich um eine Ethernet-Netzwerk-karte handelt und wo das Modul abgespeichert ist (***lib/modules/Versionsnummer/\****). Der Chipsatz ist von RealTek. Wir sehen den Namen und die E-Mail-Adresse des Modulautors und die Lizenzangaben. Des weiteren sind die konfigurierten Parametereinstellungen mit angegeben.

### Übung

Lassen Sie sich für einige Module die Informationen des Moduls mit `modinfo` anzeigen.

Das Programm ***modprobe*** ist ein highlevel-Tool für das Modul-Management.

**Tabelle 107: Optionen für modprobe**

<b>Optionen</b>	<b>Bedeutung</b>
-a	Lädt alle passenden Module.
-c	Zeigt die gerade verwendete Konfiguration.
-C DATEI	Verwendet die Datei DATEI anstelle /etc/modules.conf (oder /etc/conf.modules) für die Konfiguration.
-k	Setzt das autoclean-Flag.
-l	Listet passende Module auf.
-n	Führt keine Aktion aus.
-r	Entfernt das Modul.
-s	Sendet alles zum syslogd.
-v	Verbose.

```
#modprobe -r usb-uhci
```

Dieser Befehl entfernt das Modul usb-uhci aus dem Kernel.

```
#modprobe usb-uhci
```

Mit diesem Befehl können wir das Modul in das System aufnehmen.

Mit dem Befehl **depmod** kann man Modulabhängigkeiten in einer Art Makefile erstellen lassen, die vom Tool modprobe verwendet wird, um automatisch die richtigen Module zu laden.

Dieser Befehl wird vorrangig dazu verwendet, eigene Module in den Kernel integrierbar zu machen.

**Tabelle 108: Optionen für depmod**

<b>Optionen</b>	<b>Bedeutung</b>
-a	Sucht nach Modulen in allen Verzeichnissen, die in der Datei <code>/etc/modules.conf</code> spezifiziert sind.
-A	Überprüft nur den Zeitstempel.
-e	Zeigt alle ungelösten Probleme für jedes Modul.
-n	Schreibt das Abhängigkeitsfile auf die Standard-Ausgabe anstelle von <code>lib/modules/Version/dep.conf</code> .
-r	Notwendig, wenn Module nicht als root kompiliert wurden.
-s	Schickt alles zum syslogd.
-C DATEI	Es wird DATEI anstelle von <code>/etc/modules.conf</code> als Konfigurationsdatei verwendet.

Die Datei `/etc/modules.conf` ist die Konfigurationsdatei für ladbare Module. Was in dieser Datei passiert, wollen wir uns exemplarisch an der Netzwerkkarte bzw. am dazugehörigen Modul ansehen.

```
alias eth0 8139too
```

Diese Zeile bindet das Modul `8139too` an den Aliasnamen `eth0`, der die erste Netzwerkkarte repräsentiert. Wenn eine zweite Netzwerkkarte in dem System-Vorhanden wäre, könnten wir z. B. noch den folgenden Eintrag hinzufügen:

```
alias eth1 ne2k-pci
```

Erst dann ist die zweite Netzwerkkarte auch nutzbar.

## 7.2

### Neue Software braucht das Land

Eine wichtige Tätigkeit des Systemadministrators ist es, neue Software auf das System aufzuspielen bzw. vorhandene Software zu aktualisieren. Ein Update der Software wird sicherlich häufi-

ger vorkommen als eine Neuinstallation. Alleine das Aufkommen von diversen Sicherheitslücken kann ein Update notwendig machen. Dieser Vorgang wird auch „patchen“ genannt.

Jedes Programm auf einem System, sei es jetzt Windows oder Linux, besitzt und benötigt Programmbibliotheken. Diese Programmbibliotheken können entweder direkt (**statisch**) in das jeweilige Programm hineinkompiliert (verlinkt) sein oder auch in **dynamisch** ladbare Funktionen sogenannten **shared Libraries** verlinkt sein. Die statisch verlinkten Programme haben den Nachteil, dass sie wesentlich größer sind, da ja die Bibliothek hinein kompiliert wird. Ein weiterer Vorteil der **shared Libraries** gegenüber der statischen Version ist: wenn man mehrere Programme auf einem System hat, welche alle diese eine Bibliothek benutzen wollen, dann benötigt man diese Bibliothek nur einmal auf dem System und nicht in jedem Programm.

Der Pfad, in dem das System beim Aufruf eines Programms nach shared libraries sucht, wird in der Umgebungsvariablen **LD\_LIBRARY\_PATH** hinterlegt. Um herauszufinden, ob und welche shared libraries ein Programm verwendet, setzt man den Befehl **ldd** ab.

```
#ldd /bin/ls
    libtermcap.so.2 => /lib/libtermcap.so.2 (0x40027000)
    libc.so.6 => /lib/tls/libc.so.6 (0x42000000)
    /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Wir sehen mit diesem Befehl, von welchen shared libraries unser Befehl **ls** abhängig ist. Warum abhängig? Wenn eine dieser Bibliotheken auf dem System nicht vorhanden ist, kann das Programm **ls** nicht ausgeführt werden.

```
#ldd /usr/bin/join
    libc.so.6 => /lib/tls/libc.so.6 (0x42000000)
    /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Des Weiteren erkennen wir, dass alle shared libraries die Endung **.so** gefolgt von einer Nummer besitzen. Anhand dieser Endung kann man shared libraries erkennen. Das statische Gegenstück dazu besitzt die Endung **.a** und eine Nummer.

Wir können davon ausgehen, dass der überwiegende Teil von Linux-Programmen nicht komplett ist, das heißt, dass sie zusätzliche Funktionen zur Laufzeit benötigen, die in den shared

libraries enthalten sind. Das Tool **ld** (link loader) ist nun dafür zuständig, die von einem Programm benötigten shared libraries zu laden und das Programm zu starten.

Ferner werden auch die Verzeichnisse, die in der Datei **/etc/ld.so.conf** hinterlegt sind, durchsucht.

```
#cat /etc/ld.so.conf
/usr/kerberos/lib
/usr/X11R6/lib
/usr/lib/sane
/usr/lib/qt-3.1/lib
/usr/lib/mysql
```

Der Befehl **ldconfig** generiert die notwendigen Links und Cache, die vom link loader ld benutzt werden. Der Befehl ldconfig überprüft die Header und den Dateinamen der shared libraries, um zu definieren, welche Links einen update benötigen. Den Cache für die shared libraries findet man in der Datei **/etc/ld.so.cache**.

**Tabelle 109: Optionen für ldconfig**

<b>Optionen</b>	<b>Bedeutung</b>
-n	Überprüft nur die auf der Kommandozeile angegebenen Verzeichnisse.
-N	Baut den Cache nicht neu auf. Links werden aber neu geschrieben.
-X	Updated die Links nicht.
-f DATEI	Verwendet DATEI anstelle von /etc/ld.so.conf.
-C DATEI	Verwendet DATEI anstelle von /etc/ld.so.cache.
-r VERZ	Wechselt zu VERZ als root-Verzeichnis.
-p	Gibt den aktuellen Cache aus.

-V	Version.
-v	Gibt die Versionsnummer, den Namen der Verzeichnisse an, wie sie gescannt worden sind.

```
#ldconfig -v
/lib:
  libcap.so.1 -> libcap.so.1.10
  libnss_wins.so -> libnss_wins.so.2
  libiw.so.25 -> libiw.so.25
  libnss_winbind.so -> libnss_winbind.so.2
  libnss_ldap.so.2 -> libnss_ldap-2.3.1.so
  libpam_misc.so.0 -> libpam_misc.so.0.75
  libpam.so.0 -> libpam.so.0.75
  libproc.so.2.0.11 -> libproc.so.2.0.11
  libpamc.so.0 -> libpamc.so.0.75
  liblvm-10.so.1 -> liblvm-10.so.1.0
  libtermcap.so.2 -> libtermcap.so.2.0.8
  libpcrc.so.0 -> libpcrc.so.0.0.1
  libgcc_s.so.1 -> libgcc_s-3.2.2-20030225.so.1
  libcrypto.so.4 -> libcrypto.so.0.9.7a
  libssl.so.4 -> libssl.so.0.9.7a
  libacl.so.1 -> libacl.so.1.0.3
  libuuid.so.1 -> libuuid.so.1.2
  libattr.so.1 -> libattr.so.1.0.1
  libss.so.2 -> libss.so.2.0
  . . .
```

Nach der Installation einer neuen Library wird mit folgendem Befehl der shared library link in /lib korrekt gesetzt:

```
#ldconfig -n /lib
```

Die wichtigsten Arten Software zu installieren sind: das Installieren über die Quellen, das Installieren mittels Paket-Management-Tools von Red Hat und Debian. Wobei das von Red Hat von fast allen Distributionen verwendet wird und somit sicherlich eine dominante Stellung einnimmt.



## 7.3

**Aus der Quelle schöpfen**

Zunächst wollen wir uns dem Installieren aus den Quelldateien zuwenden. Meistens liegen uns die Quellen eines Programms in einem sogenannten tar-Ball vor. Wir erinnern uns an die Kernel-Quellen. Hier hatten wir einen komprimierten tar-Ball vorliegen. Damals verwendeten wir das Tool `gzip`. Da das Tool `gzip` nicht das einzige Komprimierungstool unter Linux ist, betrachten wir nun kurz weitere wichtige Kompressionstools. Neben `gzip` werden oft die Programme **`gunzip`** und **`zcat`** erwähnt.

Der Befehl `gunzip` dekomprimiert eine Liste von Dateien, die die Endung `.gz`, `-gz`, `.Z`, `-Z`, `_Z` oder `.Z` besitzen. Der Befehl `gunzip` erkennt auch die Kurzformen `.tgz` für `.tar.gz` und `.taz` für `.tar.Z`. Mit `gunzip` können Dateien dekomprimiert werden, die mit den Programmen `gzip`, `zip`, `compress` oder `pack` erzeugt worden sind.

**Tabelle 110: Optionen für `gunzip`**

<b>Optionen</b>	<b>Bedeutung</b>
<code>-C</code>	Schreibt auf die Standard-Ausgabe und lässt die eigentlichen Dateien unverändert.
<code>-f</code>	Erzwingt das Komprimieren und Dekomprimieren unabhängig davon, ob Dateien bereits existieren.
<code>-l</code>	Listet die Daten für komprimierte Größe, unkomprimierte Größe, Kompressionsrate und unkomprimierten Namen auf.
<code>-n</code>	Bei der Kompression wird nicht der Originalname und Zeitstempel gespeichert.
<code>-N</code>	Standard. Bei der Kompression wird der originale Dateiname und Zeitstempel verwendet.
<code>-r</code>	Rekursive Ausführung (für Directories).
<code>-t</code>	Überprüft die Kompressionsintegrität

Der Befehl `zcat` ist identisch mit dem Befehl `gunzip -c`.

**Tabelle 111: Optionen für `zcat`**

<b>Optionen</b>	<b>Bedeutung</b>
-f	Erzwingt das Komprimieren und Dekomprimieren unabhängig davon, ob Dateien bereits existieren.

Wir können nun unseren Blick weiterschweifen lassen und kommen zu einem neueren, sehr leistungsfähigen Kompressionsstool dem **`bzip2`**.

Beim Programm `gzip` verwendet man zum Komprimieren den **Lempel-Ziv** Algorithmus, und `bzip2` verwendet dafür den **Burrows-Wheeler Block Sorting** Algorithmus. Jede von `bzip2` komprimierte Datei bekommt automatisch die Endung **`.bz2`**.

**Tabelle 112: Optionen für `bzip2`**

<b>Optionen</b>	<b>Bedeutung</b>
-c	Komprimiert oder dekomprimiert auf die Standard-Ausgabe.
-d	Dekomprimiert.
-Z	Gegenteil zu -d.
-t	Testet die Integrität.
-f	Überschreibt existierende Ausgabe-dateien.
-k	Behält die Ausgangsdatei (löscht nicht).
-q	Quiet. Es werden keine Warnmeldungen ausgegeben.
-v	Verbose.
-1 bis -9	Kompressionsgüte -1 schnell, -9 beste.

Erstellen wir einmal ein tar-Archiv von unserem Verzeichnis /etc und komprimieren es anschließend mit dem bzip2-Tool.

```
#tar -cvf /etc/backup.tar /etc
. . .
#bzip2 /tmp/backup.tar
#ls -l /tmp/backup*
-rw-r--r- 1 root root 1784449 8.Sep 10:48 /tmp/backup.tar.bz2
#
```

Um dieses komprimierte File wieder zu dekomprimieren, können wir bzip2 mit der Option -d verwenden oder auch wahlweise **bunzip2** oder **bzcat**. Diese sind, wie von Ihnen schon richtig vermutet, die Gegenstücke zu gunzip und zcat.

**Tabelle 113: Optionen für bunzip2**

<b>Optionen</b>	<b>Bedeutung</b>
-f	Überschreibt existierende Ausgabe-dateien.
-k	Behält die Ausgangsdatei (löscht nicht).
-v	Verbose.

Der Befehl bzcat gibt die unkomprimierte Datei auf der Standard-Ausgabe aus.

Den Befehl tar für das Tape-Archiv haben wir bereits mit den wichtigsten Optionen kennengelernt. Wenn wir uns an das Backup vom Verzeichnis /etc erinnern und das Archiv mit der Option -t betrachten, erkennen wir, dass das führende / bei den Pfadangaben der beinhalteten Dateien fehlt. Dies ist eine Sicherheitsvorkehrung, denn stellen Sie sich vor, Sie bekommen ein Archiv und Sie entpacken es aus Versehen, ohne es vorher kontrolliert zu haben. Wenn dieses Archiv absolute Pfade beinhalten würde, würde Ihr gesamtes /etc-Verzeichnis überschrieben werden. Was das bedeutet, brauche ich nicht weiter zu erwähnen.

**Tabelle 114: Optionen für tar**

<b>Optionen</b>	<b>Bedeutung</b>
-r	Hängt an das Archiv an.
-d	Findet Unterschiede zwischen Archiv und Dateisystem.
-C	Generiert ein neues Archiv.
-t	Listet Archivinhalt auf.
-u	Fügt nur Dateien zum Archiv dazu, die neuer sind als die Copie im Archiv.
-x	Extrahiert Dateien aus dem Archiv.

Des Weiteren sind noch folgende Optionen möglich:

**Tabelle 115: Optionen für tar**

<b>Optionen</b>	<b>Bedeutung</b>
--atime-pre-serve	Verändert die AccessTime nicht.
-C DIR	Wechselt zum Verzeichnis DIR.
--checkpoint	Listet Directoryname auf, während vom Archiv gelesen wird.
-f Datei	Benutzt Datei anstelle des Standard Tape Drives /dev/st0.
-G	Generiert ein inkrementelles Backup.
-h	Speichert anstelle von symbolischen Links die Datei, auf die der Link verweist.
-k	Behält die alten Dateien.
-m	Extrahiert die Modifikationszeit nicht.

-M	Das Archiv wird über mehrere Medien verteilt.
-N Datum	Speichert nur Dateien, die jünger als Datum sind.
-p	Extrahiert alle File Permissions.
-P	Entfernt den führenden / nicht.
--remove-files	Löscht Dateien nach dem Archivieren.
--same-owner	Generiert und extrahiert Dateien mit dem gleichen Eigentümer.
-W	Überprüft das Archiv, nachdem es geschrieben wurde.
-Z	Compress und Uncompress.
-z	Gzip und gunzip.
-j	Bzip2.

Jetzt sind wir richtig präpariert, um neue Software zu installieren. In den meisten Fällen wird die neue Software direkt aus dem Internet bezogen. Wir entpacken den meist komprimierten tar-Ball in einer der oben beschriebenen Formen.

Wir spielen dies an dem führenden Web-Server, dem Apache, durch. Wir organisieren uns die Quelldateien von der Web-Site ***www.apache.org***. Die Datei (***apache\_1.3.28.tar.gz***) speichern wir in unserem HOME-Verzeichnis /root.

Wir haben zwei Möglichkeiten, die Quellen zu extrahieren.

Entweder:

```
#gzip -d apache_1.3.28.tar.gz
#tar xvf apache_1.3.28.tar
```

Oder mit der Kurzform:

```
#tar zxvf apache_1.3.28.tar.gz
```

Danach finden wir ein Verzeichnis mit dem Namen `apache_1.3.28` vor. Wir wechseln in dieses Verzeichnis und sehen uns mit `ls -l` einmal darin um.

Wir erkennen sofort ein Programm mit dem Namen ***configure***. Neben diesem Programm fallen uns auch noch die Dateien ***README*** und ***INSTALL*** auf. Wir können in diesen Dateien nachlesen, wie der Web-Server zu installieren ist und welche Optionen und Anpassungen vorgenommen werden sollen.

Wir wollen hier nicht auf die Eigenheiten des Web-Servers eingehen und überall die Standardeinstellungen verwenden. Das Programm `configure` erstellt ein ***Makefile***.

```
#./configure
Configuring for Apache, Version 1.3.28
+ Warning: Configuring Apache with default settings.
+ This is probably not what you really want.
+ Please read the README.configure and INSTALL files
+ first or at least run './configure --help' for
+ a compact summary of available options.
+ using installation path layout: Apache (con-
fig.layout)
Creating Makefile
Creating Configuration.apaci in src
```

Dieses Makefile definiert verschiedene Targets. Wir haben die Targets und den Befehl `make` bereits beim Kernel-kompilieren kennengelernt. Die Targets in dem Makefile werden mit dem Befehl `make` angestoßen. Im Normalfall genügt die Eingabe von `make` alleine, um das gesamte Programm übersetzen, also kompilieren zu lassen.

```
#make
==> src
make[1]: Wechsel in das Verzeichnis Verzeichnis
Â»/root/apache_1.3.28Â«
make[2]: Wechsel in das Verzeichnis Verzeichnis
Â»/root/apache_1.3.28/srcÂ«
==> src/regex
==> src/os
gcc -c -I../os -I../include -DLINUX=22 -DUSE_HSREGEX -
DNO_DL_NEEDED `../..apaci` os.c
```

```
gcc -c -I../os -I../include -DLINUX=22 -DUSE_HSREGEX -
DNO_DL_NEEDED `../apaci` os-inline.c
rm -f libos.a
ar cr libos.a os.o os-inline.o
ranlib libos.a
<=== src/os
==> src/ap
gcc -c -I../os -I../include -DLINUX=22 -DUSE_HSREGEX -
DNO_DL_NEEDED `../apaci` ap_cpystn.c
```

Abschließend muss das übersetzte Programm lediglich noch in das dafür vorgesehene Verzeichnis kopiert und die Zugriffsrechte entsprechend gesetzt werden. Dies geschieht durch folgende Befehlszeile:

```
#make install
make[1]: Wechsel in das Verzeichnis Verzeichnis
Â»/root/apache_1.3.28Â«
==> [mktree: Creating Apache installation tree]
./src/helpers/mkdir.sh /usr/local/apache/bin
./src/helpers/mkdir.sh /usr/local/apache/bin
./src/helpers/mkdir.sh /usr/local/apache/icons
./src/helpers/mkdir.sh /usr/local/apache/cgi-bin
. . .
. . .
[PRESERVING EXISTING CONFIG FILE: /usr/local/apache/conf/magic]
<=== [config]
make[1]: Verlassen des Verzeichnisses Verzeichnis
Â»/root/apache_1.3.28Â«
+-----+
| You now have successfully built and installed the |
| Apache 1.3 HTTP server. To verify that Apache actually |
| works correctly you now should first check the |
| (initially created or preserved) configuration files |
| |
| /usr/local/apache/conf/httpd.conf |
| |
| and then you should be able to immediately fire up |
| Apache the first time by running: |
| |
| /usr/local/apache/bin/apachectl start |
| |
| Thanks for using Apache. The Apache Group |
| http://www.apache.org/ |
+-----+
```

Damit haben wir einen Web-Server auf unserem System installiert. Wir können diesen Web-Server starten, indem wir folgende Befehlszeile eingeben:

```
#/usr/local/apache/bin/apachectl start
```

Die Fehlermeldung, dass kein Servername gefunden werden konnte, ignorieren wir einfach. Sie kommt daher, da wir keinen Namens-Dienst verwenden können. Auf das Thema Namens-Dienst bzw. Name-Service oder DNS kommen wir im Kapitel 14.1 noch genauer zu sprechen.

Wir können die korrekte Funktion unseres Web-Servers aber dennoch gleich austesten, indem wir ein Browserfenster, entweder den Konqueror, den Mozilla, den Opera oder sonst einen unter Linux erhältlichen Browser, öffnen und im URL-Feld localhost eingeben und Return drücken. Wenn uns kein Fehler unterlaufen ist, bekommen wir auch keine Fehlermeldung, sondern eine HTML-Seite des Apache Servers angezeigt, die uns willkommen heißt.

Damit haben wir die Softwareinstallation auf traditionelle Art und Weise durchgespielt. Manche Programme haben in ihrem Makefile auch ein Target für das Deinstallieren. Dieses Target kann man mittels folgenden Befehl ausführen lassen:

```
#make uninstall
```

## 7.4

### Alle wollen Red Hat-Pakete

Findige Leute haben sich zusammengesetzt und sich Gedanken gemacht, wie man den Prozess des Softwareinstallierens vereinfachen und komfortabler gestalten kann. Herausgekommen ist das sogenannte **rpm**-Management (Red Hat Package Management). Es gab neben rpm auch noch andere, allerdings hat sich rpm bei fast allen Distributionen durchgesetzt, und somit verwenden fast alle diese Art des Paket-Managements.

Bei einem Paket-Management wird die zu installierende Software, die Konfigurationsdateien und alle zugehörigen Dateien in einem Paket, dem sogenannten Package, zusammengefaßt, und das Paket-Management kann Pre- und Post-Install-Skripts ausführen. Des Weiteren steht uns mit einem Managementsystem auch ein sehr mächtiges Abfrage- und Verwaltungssystem zur Verfügung, das es uns sehr erleichtert, Pakete zu installieren, upzudaten und wieder zu deinstallieren.



Wir wollen uns dies am Beispiel des Opera-Browsers ansehen. Wir laden die freie Trialversion von ***www.opera.com*** in unser HOME-Verzeichnis.

#### *Hinweis*

Wir verwenden die statische Version des Opera-Browsers, damit wir mit den verwendeten Libraries keine Probleme bekommen. Denn Opera verwendet Libraries, die nicht im Standardumfang unserer installierten Red Hat-Version enthalten sind. Wir wollen aber nicht alle Bibliotheken herunterladen und installieren. Ich empfehle Ihnen dies aber in der Prüfungsvorbereitung zu versuchen, da hierbei alle Konzepte der shared libraries geübt werden können.

Nun ist alles ganz einfach:

```
#rpm -i -nodeps opera-7.11-20030515.1-static-qt.i386.rpm
```

Den Parameter `-nodeps` benötigen wir hier nur, weil in dem Paket der Operaversion eine falsche, nicht notwendige Abhängigkeit eingebaut ist.

Nun können wir den Opera-Browser durch folgende Kommandozeile als Hintergrundprozess starten.

```
#opera &
```

**Tabelle 116: Optionen für rpm**

<b>Optionen</b>	<b>Bedeutung</b>
<code>-v</code>	Gibt Statusmeldungen aus. Default ist quiet.
<code>--pipe</code>	Piped die Ausgabe von rpm an das folgende Kommando.
<code>--dbpath DIR</code>	Verwendet DIR anstelle von <code>/var/lib/rpm</code> als DB-Pfad.
<code>-i</code>	Installiert das Paket.
<code>-U</code>	Upgrade.
<code>-F</code>	Macht das Upgrade nur, wenn bereits aktuell dieses Paket installiert ist.

-e	Erase. Löscht das angegebene Paket.
-q	Query. Abfrageoption.
--initdb	Baut die Datenbank wieder neu auf.

Zusätzlich zu diesen Optionen existieren noch folgende Parameter. (Es folgt eine Auswahl der wichtigsten, da der Umfang ansonsten erdrückend wäre):

**Tabelle 117: Weiter rpm Optionen**

<b>Parameter</b>	<b>Bedeutung</b>
--allfiles	Installiert oder upgradet alle Dateien in dem Paket unabhängig davon, ob sie existieren oder nicht.
--ignoresize	Testet die gemounteten Filesysteme nicht, ob genügend Platz vorhanden ist.
--includedocs	Installiert Dokumentationsdateien (Default).
--nodeps	Führt keinen Abhängigkeitstest durch.
--nosignature	Überprüft die digitale Signatur nicht.

Das rpm-Management stellt uns auch digitale Signaturen zur Verfügung. Mit dem Befehl `rpm -import PUBKEY` wird die digitale Signatur geladen.

`#rpm -checksig Paketfile`

Damit kann die digitale Signatur eines Paketes kontrolliert werden. Ohne einen öffentlichen Schlüssel funktioniert dies allerdings nicht. Digitale Signaturen können nur mit einem öffentlichen Schlüssel verifiziert und abgefragt werden.

Wenn man eine digitale Signatur zu einem Paket hinzufügen möchte, verwendet man folgende Befehlszeile:

```
#rpm -addsig Package
```

Für die Query-Option existieren noch folgende zusätzlichen Parameter:

**Tabelle 118: Parameter für die rpm Query Option**

<b><i>Parameter</i></b>	<b><i>Bedeutung</i></b>
-a	Alle installierten Pakete.
-f DATEI	Listet das Paket auf, zu dem DATEI gehört.
-p PACK	Fragt ein uninstalliertes Paket PACK ab.
-c	Listet nur Konfigurationsdateien auf.
-d	Listet nur Dokumentationsfiles auf.
-l	Listet Dateien der Pakete auf.

Wenn uns interessiert, welche Pakete in unserem System bereits installiert sind, verwenden wir Folgendes:

```
#rpm -qa
setup-2.5.25-1
bzip2-libs-1.0.2-8
e2fsprogs-1.32-6
glib-1.2.10-10
iputils-20020927-2
losetup-2.11y-9
net-tools-1.60-12
shadow-utils-4.0.3-6
libtermcap-2.0.8-35
MAKEDEV-3.3.2-5
raidtools-1.00.3-2
hotplug-2002_04_01-17
file-3.39-9
```

```
findutils-4.1.7-9
```

```
. . .
```

Wenn wir uns dafür interessieren, ob der Opera Web-Browser installiert ist, können wir folgende zwei Befehle bemühen:

```
#rpm -qa | grep opera
opera-7.11-20030515.1
#
#rpm -q opera
opera-7.11-20030515.1
```

Um herauszufinden, ob Dokumentationsfiles und Konfigurationsdateien für den Opera vorhanden sind und wie diese heißen, geben wir dies ein:

```
#rpm -qd opera
/usr/share/doc/opera/LICENSE
/usr/share/doc/opera/help
#
#rpm -qc opera
/etc/opera6rc
/etc/opera6rc.fixed
#
```

Eine sehr elegante Art, Pakete zu installieren, ist über das Netzwerk z. B. mit **ftp** (file transfer protocol).

```
#rpm -i ftp://ftp.installserver.at/pub/neues.rpm
```

Wenn man feststellen möchte, zu welchem Paket die Datei /sbin/modprobe gehört, verwendet man Folgendes:

```
#rpm -qf /sbin/modprobe
modutils-2.4.22-8
```

Um Informationen über unser Opera-RPM zu erhalten, bevor wir es installieren, verwenden wir folgenden Befehl:

```
#rpm -qpi opera-7.11-20030515.1-static-qt.i386.rpm
Name       : opera       Relocations: (not relocateable)
Version    : 7.11
Vendor:    Opera Software ASA
Release    : 20030515.1
Build Date: Don 15 Mai 2003 18:34:26 CEST
Install Date: (not installed)
```

```
Build Host: palantir.intern.opera.no
Group      : Applications/Networking
Source RPM: opera-7.11-20030515.1.src.rpm
Size       : 12158724
License: Commercial
Signature  : (none)
Packager   : Christian Westgaard <chrisw@opera.com>
Summary    : Opera, statically linked to Qt 3.0.5-mt
Description: Opera 7.11
```

Welcome to the Opera Web browser. It is smaller, faster, customizable, powerful, yet user-friendly. Opera eliminates sluggish performance, HTML standard violations, desktop domination, and instability. This robust Web browser lets you navigate the Web at incredible speed and offers you the best Internet experience.

The binaries was built on a Red Hat 6.2 installation using gcc-2.95.3.

```
#rpm --showrc
```

Dieser Befehl zeigt alle Werte an, die rpm aktuell verwendet bzw. in den Dateien **/usr/lib/rpm/rpmrc** und **macros** Konfigurationsdateien gesetzt hat.

Um den Opera letztendlich wieder von unserem System zu entfernen, verwenden wir einfach den Befehl:

```
#rpm -e opera
```

## 7.5

### **Etwas zum Genießen – Die Debian-Paketverwaltung**

Das Debian Paket-Management-System ist sicherlich eines der leistungsfähigsten und komfortabelsten. Leider steht es uns nur unter der Debian-Distribution zur Verfügung.

Das Debian-Paket-Format kennt unterschiedliche Bezeichnungen. Dabei ist nicht die Endung der Pakete gemeint, die immer **.deb** ist, sondern vielmehr der Status der Pakete.

Der erste Status, in dem sich ein Paket befinden kann, ist **unstable**. Hier sind Pakete gemeint, die sich gerade in Entwicklung bzw. in Weiterentwicklung befinden. Dass es hier auch zu größeren Fehlern in den Paketen kommen kann, versteht sich, glaube ich, von selbst.

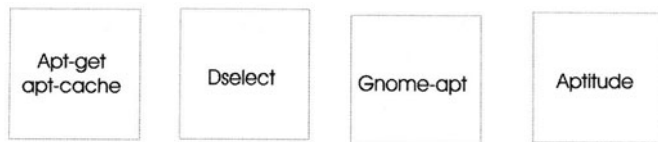
Täglich werden aus der unstable die **testing**-Pakete und damit die testing-Distribution erzeugt. Die Regeln, die dahinter stehen, liefern schon eine sehr aktuelle und stabile Distribution.

Wenn die Entwickler der einzelnen Pakete mit der Stabilität der testing-Distribution zufrieden sind, wird aus dieser die sogenannte **stable**-Version erzeugt.

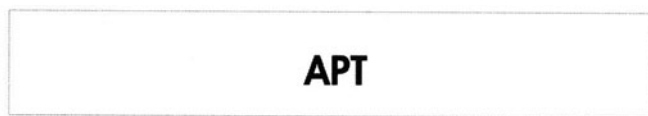
Es wird schließlich unter dem **main**-Teil, den **non-free**-Teil und dem **contrib**-Teil unterschieden. Im main-Teil sind alle freien Pakete, im non-free-Teil sind Pakete, die einigen Einschränkungen unterliegen, und in der contrib sind Pakete, die eigentlich frei sind aber non-free-Pakete voraussetzen.

Das Debian Paketverwaltungs-System besteht aus 4 kleinen Tools.

Frontend



Mittelschicht



Backend



Abbildung 12: Das Debian Paket-Managemetsystem

Das Herz der Paketverwaltung bildet das Programm **dpkg**. Es ist vergleichbar mit dem rpm-Tool von Red Hat und wird mit der Datei `/etc/dpkg/dpkg.cfg` konfiguriert.

Nachdem, im Gegensatz zu anderen Distributionen, die Debian-Pakete hauptsächlich von Debian selbst kommen, gab es schon sehr früh eine Liste aller Debian-Pakete. In dieser Liste sind alle Pakete, Paketabhängigkeiten und Downloadseiten gespeichert.

Mit dem Tool **apt** werden diese Pakete geholt. Damit kennt apt alle verfügbaren Pakete und mit Hilfe von dpkg, auch alle installierten. Wir sagen zu dem apt-Tool: installiere das Paket xy. Dann kann apt dieses direkt aus dem Internet laden und mit Hilfe von dpkg installieren. Nachdem aber apt auch andere Informationen des Paketes kennt, wie z. B. Paketabhängigkeiten, kann es auch andere benötigte Pakete holen und installieren oder entfernen. Das Tool apt kann nicht nur mit dem Internet umgehen, sondern auch mit anderen Quellen wie CD-ROM oder Installations-Server.

Betrachten wir nun erst einmal das Herz der Paketverwaltung dpkg, auch wenn wir es zu allererst gar nicht direkt brauchen werden.

Mit der folgenden Befehlszeile wird das Paket icewm\_0.8.12-1.deb installiert.

```
#dpkg -i icewm_0.8.12-1.deb
```

Wenn man bereits eine ältere Version des Programms icewm installiert hat, wird dpkg dies automatisch aktualisieren anstelle von zwei Versionen des Programms zu installieren.

Wollen wir das Programm icewm wieder von unserem System deinstallieren, können wir das auf zwei Arten erledigen:

```
#dpkg -r icewm
```

oder:

```
#dpkg --purge icewm
```

Der Unterschied liegt darin, dass bei der zweiten Art auch die Konfigurationsdateien wieder gelöscht werden. Uns ist dabei aufgefallen, dass wir nur den Namen des Programms angeben müssen - im Gegensatz zum Installieren, wo der gesamte Paketname anzugeben ist.

Um eine vollständige Liste aller auf dem System installierten Pakete zu erhalten, geben wir folgenden Befehl ein:

```
#dpkg -l
```

Das Tool apt ist die sogenannte Mittelschicht, und alle anderen Programme der Frontends setzen auf diesem Tool auf. Dieses Tool wird über Konfigurationsdateien definiert. Diese Konfigurationsdateien sind im Verzeichnis **/etc/apt** zu finden. Hierin findet man bis zu drei Konfigurationsdateien.

Damit apt die Pakete holen kann, benötigt es Informationen über die Quellen bzw. den Ort, von wo die Pakete geholt werden können. Diese Informationen sind in der Konfigurationsdatei **/etc/apt/sources.list** abgelegt. In dieser Datei wird pro Zeile eine Quelle aufgeführt. Das Format sieht folgendermaßen aus:

Typ URI Distribution Komponent1 Komponente 2 ...

Als **Typ** kann man **deb** für binary Quellen und **deb-src** für Sourcequellen angeben.

Als URI-Typen können **file**, **http**, **ftp**, **cdrom** und die Adresse angegeben werden.

#### Hinweis

Bitte beachten Sie, dass die oft fälschlicherweise synonym verwendeten Begriffe URI und URL unterschiedlich sind. Die URL ist z. B. [www.lpi.org](http://www.lpi.org). Im Gegensatz dazu ist die entsprechende URI gegeben durch <http://www.lpi.org>, also ergänzt mit der Protokollangabe. Als Distribution kann z. B. **stable**, **testing**, **unstable** angegeben werden. Abschließend können noch die Komponenten **main**, **contrib**, etc. angegeben werden.

So könnte ein Eintrag in der Datei **sources.list** folgendermaßen aussehen:

```
deb file:/home.jason/debian stable main contrib non-free
deb ftp://ftp.debian.org/archive stable main contrib
```

Zusätzlich kann der Systemadministrator in der Konfigurationsdatei **preferences** noch genau festlegen, welche Pakete aus welcher Distribution mit welcher Version verwendet werden sollen. Nachdem wir in der **sources.list** Quellen verschiedener Distributionen eingetragen haben, tragen wir noch in die Konfigurationsdatei **apt.conf** folgende Zeile ein:

```
APT::Default-Release {"stable" ;};
```

Damit konfigurieren wir das apt-Tool so, dass es generell nur Pakete aus der **stable**-Distribution betrachten soll.

Nun wollen wir uns die sogenannten Frontends zum apt-Tool ansehen.

Mit **apt-cache** kann man die apt-Datenbank abfragen. Wir können mit dem apt-cache-Tool z. B. die Datenbank nach einem bestimmten Paket durchsuchen.

```
#apt-cache search samba
```

Dies sucht nach dem Paket **samba**.



**Tabelle 119: Optionen für apt-cache**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
search	Sucht nach Einträgen in der Datenbank. Mit den Parametern -names-only werden nur die Paketnamen durchsucht und mit -full alle Informationen.
show	Zeigt alle Informationen zum angegebenen Paket.
policy	Hierbei erhält man eine Übersicht aller Versionen eines Paketes, die installierte Version und die Version, die installiert wurde.

Das nächste Tool ist ***apt-get***.

```
#apt-get install less
```

Diese Kommandozeile würde das Paket less downloaden und installieren.

**Tabelle 120: Optionen für apt-get**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
install	Installiert das angegebene Paket. Hängt man bei einem Konflikt ein - an das Paket an, wird es vom System gelöscht, mit der Option -f kann man ein Problem von apt-get selbst lösen lassen, und mit -reinstall kann man es noch einmal installieren lassen. Mit der Option -d kann man die Pakete laden lassen, ohne sie gleich zu installieren.

update	Lädt die Beschreibungen der Pakete neu und updatet die Packages.gz-Dateien auf dem System. Danach stehen die Informationen allen Frontends zur Verfügung.
upgrade	Damit werden alle derzeit installierten Pakete bzw. Programme aktualisiert. Mit der zusätzlichen Option -s kann man sich anzeigen lassen, was getan wird, und mit -d kann man die Pakete nur herunterladen lassen, ohne sie zu installieren.
remove	Es können damit Pakete gelöscht werden. Mit der Option -purge werden auch die Konfigurationsdateien gelöscht.
check	Überprüft die Installation auf defekte Pakete.
dist-upgrade	Wie upgrade.
clean	Alle Pakete, die installiert wurden, sind vorher in das Verzeichnis /var/cache/apt/archives geladen worden. Dies kann unter Umständen viel Platz einnehmen. Die Option clean löscht nun die zwischengespeicherten Dateien.
autoclean	Löscht nur die Pakete, die nicht mehr auf den Debian-Quellen verfügbar sind.
source	Damit kann man die Quellen der einzelnen Pakete installieren.

Mit dem Tool **tasksel** kann man auf sehr einfache Art und Weise mehrere Pakete für einzelne Aufgabengebiete auswählen und installieren lassen. Mit Return und Space kann man verschiedene

**Tasks** auswählen, und mit **I** bekommt man kurze Informationen dazu.

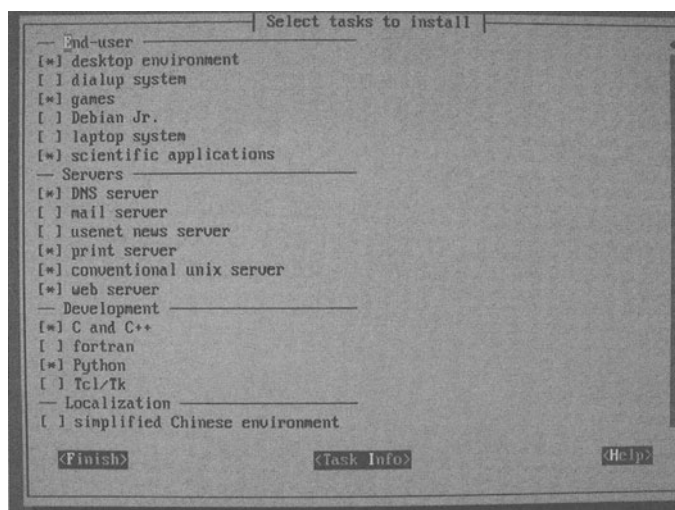


Abbildung 13: *tasksel* Startbildschirm

Mit dem Tool **dselect** kann man über eine Menüsteuerung sehr komfortabel einzelne Pakete zur Installation auswählen, die mit Hilfe von **apt-get** installiert werden.

Aufgerufen wird das Tool einfach durch die Eingabe **dselect**.

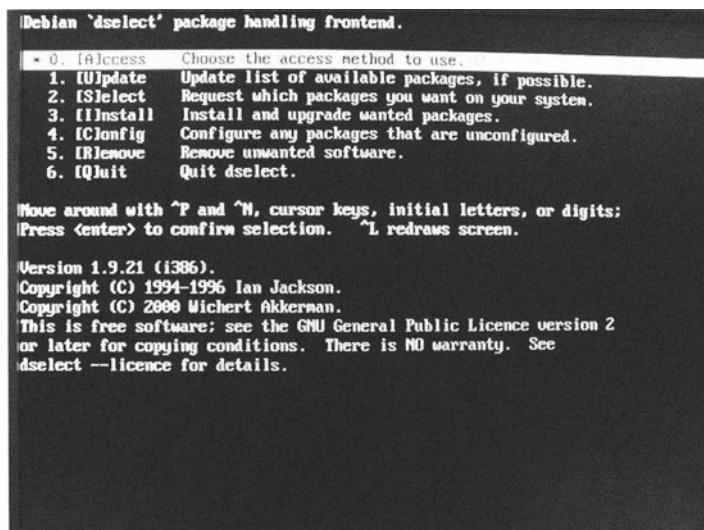


Abbildung 14: Startbildschirm vom *dselect*-Tool

```
dselect - list of access methods
Abbrev.      Description
cdrom        Install from a CD-ROM.
nfs          Install from an NFS server (not yet mounted).
harddisk     Install from a hard disk partition (not yet mounted).
mounted      Install from a filesystem which is already mounted.
floppy       Install from a pile of floppy disks.
* apt        API Acquisition [file,http,ftp]

Access method 'apt'.
apt - API Acquisition [file,http,ftp]

The APT installation method encompasses most other installation methods
under the umbrella of the new Package Acquisition code. This method allows
installation from locations in the filesystem, ftp and http URLs, supports
full installation ordering and dependency checking as well as multiple
sources. See the man pages apt-get(8) and sources.list(5)

HTTP proxies can be used by setting http_proxy="http://proxy:port/" before
running DSelect. FTP proxies require special configuration detailed in the
apt.conf(5) man page (see /usr/doc/apt/examples/apt.conf)
```

Abbildung 15: Access Abschnitt des dselect Tools

Mit **Access** definiert man, woher die Pakete kommen sollen. Der Menüpunkt **Update** besitzt die gleiche Wirkung wie das update beim apt-get Tool. **Select** stellt nun die zentrale Komponente des dselect-Tools dar. An dieser Stelle können Sie jedes einzelne Paket auswählen, das auf dem System installiert werden soll. Durch Drücken der Return-Taste wird die Auswahl akzeptiert und in das Hauptmenü zurückgekehrt. Abhängigkeiten sind normal, und dselect wird uns warnen und uns einen Lösungsvorschlag anbieten. Wenn sich zwei Pakete gegenseitig ausschließen, stehen wir vor der mühseligen Aufgabe, uns für eines von beiden zu entscheiden.

```

dselect - main package listing (avail., priority) mark:++/- verbose:u help
E10N Fri Section Package Inst.ver Avail.ver Description
- All packages -
  Newly available packages
  New Optional packages
  New Optional packages in section admin
n_ Opt admin acct <none> 6.3.5-32 The GNU Accounting uti
n_ Opt admin adjtimex <none> 1.13-1 Utility to display or
n_ Opt admin aide <none> 0.8-2 Advanced Intrusion Det
n_ Opt admin alien <none> 8.05 install non-native pac
n_ Opt admin anacron <none> 2.3-6 a cron-like program th
n_ Opt admin apmd <none> 3.0.2-1.19 Utilities for Advanced
n_ Opt admin apt-localepu <none> 0.0.37 Dummy installer packag
n_ Opt admin apt-move <none> 4.1.21 Move cache of Debian p
n_ Opt admin apt-show-sou <none> 0.06-3 Shows source-package i
All packages
The line you have highlighted represents many packages; if you ask to
install, remove, hold, &c it you will affect all the packages which match
the criterion shown.

If you move the highlight to a line for a particular package you will see
information about that package displayed here. You can use 'o' and 'O' to
change the sort order and give yourself the opportunity to mark packages in
different kinds of groups.

```

Abbildung 16: Select Abschnitt des dselect Tools

Tabelle 121: Steuerung von dselect

Tasten	Bedeutung
+	Wählt ein Paket zur Installation aus.
=	Stellt ein Paket zurück.
-	Löscht ein Paket.
-	Löscht ein Paket inkl. Konfigurationsdateien.
I, i	Schaltet das Infofenster um.
O, o	Schaltet die Sortierung um.
V, v	Ausführliche Darstellung aus/ein.

Die Pakete werden mit ihrem Zustand aufgelistet.

**Tabelle 122: Paketstatus**

<b>Zustand</b>	<b>Bedeutung</b>
E	Fehler.
I	Installationszustand.
O	Alte Markierung.
M	Aktuelle Markierung.

Wählt man anschließend den Menüpunkt **Install**, werden alle Pakete nacheinander in das System eingespielt. Mit Menüpunkt **Configure** kann man die Konfiguration der Pakete wiederholen. Normalerweise werden die Pakete bei der Installation konfiguriert. Der Menüpunkt **Remove** löscht unbenötigte Pakete vom System. Und schließlich kann man mit **Quit** wieder aus dem Tool aussteigen.

Mehrere hundert Pakete verwenden derzeit **debconf** zur Konfiguration bei der Installation der Pakete. Mit dem Tool debconf sollen nicht die herkömmlichen Konfigurationsdateien abgelöst werden, sondern ein einheitliches Frageninterface bereitgestellt werden. Wenn man nach einiger Zeit feststellt, dass die eine oder andere Antwort, die man einmal an debconf gegeben hat, sich jetzt verändert hat, kann man diese Fragen mit dem Befehl **dpkg-reconfigure** nochmals eingeben. Die entsprechenden Antworten werden dann gespeichert und zur Konfiguration verwendet.

Damit haben wir die wichtigsten Wege, neue Software auf den Rechner zu installieren, kennen gelernt. Nun wollen wir uns im nächsten Kapitel dem großen Thema Hardware widmen.

Neben der Aufgabe, neue Software zu installieren und vorhandene zu aktualisieren, ist die Installation und Inbetriebnahme neuer oder erneuerter Hardware eine weitere wichtige Aufgabe des Systemadministrators.

Wenden wir uns aber zuvor einigen grundsätzlichen Überlegungen zu. Linux ist für viele Hardwarearchitekturen portiert worden. Neben den klassischen Rechnerarchitekturen mit ISA, VLB, EISA, PCI und USB-Bussen ist es auch möglich, Linux auf einem PDA einer Playstation2 oder sogar der Xbox zu installieren. Wir wollen uns aber auf die Intel/AMD-Architektur mit ISA, PCI und USB-Bussen beschränken.

### 8.1

#### Am Anfang war das BIOS

Nach dem Einschalten des Computers nimmt zuerst das BIOS seine Arbeit auf. Bei den heutzutage gängigen Motherboards sind schon sehr viele Peripheriegeräte on-board. Das heißt, man findet Geräte wie Soundkarte, Netzwerkkarte, Grafikkarte u.s.w. nicht mehr als eigene Steckkarten im PC vor, sondern deren Funktionalitäten sind bereits auf dem Motherboard integriert. Somit ist die Einstellung bzw. das Aktivieren und Deaktivieren durch die einschlägigen Einstellungen im verwendeten BIOS vorzunehmen. Wenn ich also ein Motherboard mit Sound on-board besitze und auf eine neue Soundkarte, die ich als PCI-Steckkarte erhalten habe, umsteigen möchte, muss ich zuvor die Soundunterstützung im BIOS für das on-board-Gerät abschalten um die neue verwenden zu können, ohne Konflikte zu riskieren.

Da von verschiedenen PC-Herstellern unterschiedliche Motherboards und damit auch unterschiedliche Versionen und Anbieter von BIOS'en verwendet werden, können wir hier nicht darauf eingehen, wie die einzelnen Geräte im BIOS ein und ausgeschaltet sowie konfiguriert werden. Ich rate Ihnen deshalb, bei der aktuellen Einstellung in das BIOS-Manual Ihres Motherboards zu blicken.

## 8.2

### Hardware und das System

Linux unterstützt eine große Zahl von Hardware, welche im „**Linux Hardware Compatibility HOWTO**“ genau nachgelesen werden kann.

Standardmäßig werden von Linux die seriellen Anschlüsse (com-Ports) unterstützt. Aber auch der Parallel-Port für den Druckeranschluss und der Joystick-Port werden unterstützt. Unterstützt heißt hier, dass die notwendigen Kernel-Optionen richtig eingestellt bzw. als Modul konfiguriert worden sind. Beim Defaultkernel ist allerdings nur die Unterstützung der seriellen Interfaces gewährleistet.

Wie kommuniziert man nun mit der Hardware? Besser gesagt über welche Mechanismen und Komponenten verwaltet das verwendete Betriebssystem die angeschlossene Hardware? Die Komponenten, die zum Tragen kommen, sind:

**Interrupt**, **I/O-Port**, **I/O-Speicher** und **DMA** sind Ressourcen der Computerarchitektur, die für die Kommunikation mit der Hardware zuständig sind.

Interrupts veranlassen die CPU, ihre derzeitige Arbeit zu unterbrechen, und mit dem, was der Interrupt veranlasst, fortzufahren. Der I/O (Ein/Ausgabe)-Port ist jener Bereich, über den sich die Hardware an das System andockt. Letztendlich ist der DMA (direct memory access) dafür da, Daten direkt in den Speicher zu schreiben und nicht über den Umweg der CPU.

Damit wissen wir schon, was eigentlich die hauptsächliche Aufgabe eines Treibers ist, nämlich in den I/O-Speicher zu schreiben und zu lesen.

Warum sind nun die Informationen über IRQ, I/O-Ports und DMA wichtig, wenn wir doch ohnehin schon alle PCI-Systeme mit Plug and Play-Unterstützung besitzen und Linux mit PnP Komponenten umgehen kann? Die Antwort darauf ist ganz einfach. Wir können trotz PnP einmal in die Lage kommen, eine ISA-Karte konfigurieren oder sonst irgendwelche Hardwarekonflikte lösen zu müssen. In diesen Fällen ist es immer gut, wenn man weiß, wo man suchen und schrauben muss, und nicht nur, weil es bei der LPI-Prüfung verlangt wird.

Mit dem Befehl **setserial** kann man das Verhalten bzw die Informationen für die seriellen Geräte in einem Linux-Rechner definieren. Diese Informationen beinhalten, welcher I/O-Port und IRQ u.s.w. verwendet wird.



Bei einem normalen Bootvorgang werden nur die seriellen Ports 1 bis 4 mit den Standardeinstellungen initialisiert. Will man weitere serielle Schnittstellen oder andere Konfigurationsdaten einstellen, verwendet man das Programm `setserial`.

**Tabelle 123: Optionen für `setserial`**

<b>Option</b>	<b>Bedeutung</b>
-a	Gibt alle verfügbaren Informationen aus.
-b	Gibt eine Zusammenfassung für jedes Device aus.
-G	Erzeugt Konfigurationsausgaben, die wiederum als Argument für den Befehl <code>setserial</code> verwendet werden können.
-Z	Setzt alle Flags zurück, bevor Neue gesetzt werden.

Parameter, die mit `setserial` gesetzt werden können:

**Tabelle 124: Parameter für `setserial`**

<b>Parameter</b>	<b>Bedeutung</b>
port NUMBER	Der I/O-Port wird gesetzt.
irq NUMBER	Der IRQ wird auf NUMBER gesetzt.
uart TYPE	Der UART-Type wird gesetzt z.B: 16550A.
autoconfig	Der Kernel wird aufgefordert, die serielle Schnittstelle automatisch zu konfigurieren.
baud_base	Mit dieser Option kann die Baud-Rate gesetzt werden. Dies ist die Taktfrequenz dividiert durch 16. Normalerweise ist sie auf 115200 eingestellt.

spd_hi	Verwende 57.6kb, wenn nach 38.4 kb verlangt wird.
spd_vhi	Verwende 115kb, wenn nach 38.4kb verlangt wird.
spd_shi	Verwende 230kb, wenn nach 38.4 kb verlangt wird.
spd_normal	Verwende 38.4, wenn nach 38.4 verlangt wird.
close_delay	Spezifiziert die Zeit in 1/100 Sekunden, in der DTR low sein soll, damit das device geschlossen werden soll.
closeing_wait	Spezifiziert die Zeit in 1/100 Sekunden, in der der Kernel auf Daten warten soll, während der Port geschlossen wird.

Sehen wir uns dies nun in der Praxis an:

```
#setserial -G /dev/ttyS0
/dev/ttyS0 uart 16550A port 0x03f8 irq 4 baud_base
115200 spd_normal skip_test
#
#setserial -a /dev/ttyS0
/dev/ttyS0, Line 0, UART: 16550A, Port: 0x03f8, IRQ: 4
Baud_base: 115200, close_delay: 50, divisor: 0
closing_wait: 3000
Flags: spd_normal skip_test
#
```

Wenn wir die Geschwindigkeit, also die Baudrate auf 9600 setzen wollen, verwenden wir folgenden Befehl:

```
#setserial /dev/ttyS0 baud_base 9600
```

Dies können wir wieder mit der Option -a von setserial überprüfen. Um die Baudrate auf die Geschwindigkeit von 115200 Baud zu setzen, geben wir einfach folgenden Befehl ein:

```
#setserial /dev/ttyS0 baud_base 115200
```

**Wichtig**

Mit `setserial` kann man auch jene Einstellungen vornehmen, die für die reibungslose Verwendung eines Modems für die DFÜ-Verbindung mit dem Internet notwendig sind.

In diesem Zusammenhang ist es von großer Bedeutung, auf die Problematik von **Winmodems** hinzuweisen. Im Normalfall sind Modems externe Geräte, die an einen seriellen Port angeschlossen sind. Oft findet man aber auch Modems, die als Einsteckkarte vorhanden sind und einen weiteren seriellen Port darstellen, also **`ttyS3`** oder **`ttyS4`**. Vorsicht ist bei Laptops geboten, da hier oft sogenannte WinModems verwendet werden. Diese sind keine richtigen Modems, sondern bestehen nur aus ein wenig Hardware und werden eigentlich von einer Software als Modem emuliert. Wenn man also einen Rechner mit solch einem Modem hat, sollte man auf die Web-Site **[www.linmodems.org](http://www.linmodems.org)** gehen.

Linux unterhält ein eigenes Dateisystem, welches von Linux selbst angelegt und verwaltet wird. Dieses Dateisystem ist auf **`/proc`** gemountet und unterhält bzw. beinhaltet alle Informationen über Kernelparameter, Hardware, laufende Prozesse und Speicherbereiche.

**Tabelle 125: IRQ/DMA und I/O-Bereiche von Geräten**

<b>Gerät</b>	<b>I/O-Port</b>	<b>IRQ</b>	<b>DMA</b>
<code>ttyS0 (com1)</code>	3f8	4	--
<code>ttyS1 (com2)</code>	2f8	3	--
<code>lp0 (lpt1)</code>	378-37f	7	--
<code>lp1 (lpt2)</code>	278-27f	5	--
<code>fd0 (Floppy)</code>	3f0-3f7	6	2

Wechseln wir in das Verzeichnis `/proc`. Dort finden wir die Dateien, die Informationen über die soeben besprochenen Ressourcen beinhalten.

```
#cat /proc/interrupts
```

```

CPU0
0:   2287393      XT-PIC  timer
1:     1755      XT-PIC  keyboard
2:         0      XT-PIC  cascade
```

```

5:          0          XT-PIC  Intel ICH 82801AA
6:          7          XT-PIC  floppy
8:          1          XT-PIC  rtc
10:         11286       XT-PIC  eth0
11:          0          XT-PIC  usb-uhci
12:         91271       XT-PIC  PS/2 Mouse
14:        224826       XT-PIC  ide0
15:        450343       XT-PIC  ide1
NMI:          0
ERR:          0

```

Hier erkennen wir, dass die Maus eine PS/2-Maus ist und den Interrupt 12 besitzt. Wenn wir eine Hardwareerweiterung vornehmen wollen, die ebenfalls auf den Interrupt 12 konfiguriert ist, müssen wir diese umkonfigurieren, entweder mit den dafür vorgesehenen Jumpers auf der Karte oder über ein Software-konfigurationstool, wie es oft bei Netzwerkkarten üblich ist. Übrigens hat unsere Netzwerkkarte eth0 hier den Interrupt 10.

```

#cat /proc/ioports
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02f8-02ff : serial(auto)
0376-0376 : ide1
0378-037a : parport0
03c0-03df : vga+
03f2-03f5 : floppy
03f6-03f6 : ide0
03f7-03f7 : floppy DIR
03f8-03ff : serial(auto)
0cf8-0cff : PCI conf1
2000-20ff : Accton Technology Corporation SMC2-1211TX
          2000-20ff : 8139too

```

3000-30ff : Intel Corp. 82801AA AC'97 Audio  
3000-30ff : Intel ICH 82801AA  
3400-343f : Intel Corp. 82801AA AC'97 Audio  
3400-343f : Intel ICH 82801AA  
3440-345f : Intel Corp. 82801AA USB  
3440-345f : usb-uhci  
3460-346f : Intel Corp. 82801AA IDE  
3460-3467 : ide0  
3468-346f : ide1

Wir sehen gleich, dass unsere Netzwerkkarte den I/O-Port 2000-20ff benutzt. Die Bereichsangaben sind in hexadezimaler Form. Des Weiteren finden wir heraus, dass es sich um eine Netzwerkkarte von Accton Technology Corporation mit der Typenbezeichnung SMC2-1211TX handelt. Darunter finden wir die Angabe darüber, welcher Treiber bzw. in unserm Fall welches Kernel-Modul diesen Bereich belegt, also an welches Modul der I/O-Port gebunden ist – nämlich an 8139too, das ist der Realtec Chipsatztreiber.

```
#cat /proc/iomem
00000000-0009fbff : System RAM
0009fc00-0009ffff : reserved
000a0000-000bffff : Video RAM area
000c0000-000c7fff : Video ROM
000e0000-000effff : Extension ROM
000f0000-000fffff : System ROM
00100000-07fcffff : System RAM
    00100000-00250d5b : Kernel code
    00250d5c-0034ac43 : Kernel data
07fd0000-07feffff : ACPI Non-volatile Storage
07ff0000-07ffffff : System RAM
40400000-404000ff : Accton Technology Corporation SMC2-
    40400000-404000ff : 8139too
41000000-421fffff : PCI Bus #01
    41000000-41ffffff : nVidia Corporation RIVA TNT2 Model
46000000-47ffffff : PCI Bus #01
    46000000-47ffffff : nVidia Corporation RIVA TNT2 Model
48000000-4bffffff : Intel Corp. 82815 815 Chipset Host
Bridge and Memory Controller Hub
feea0000-ffffffff : reserved
```

Hier finden wir, dass unsere Netzwerkkarte in den I/O-Speicherbereich 40400000-404000ff schreibt. Diese Angaben sind wieder in hexadezimaler Schreibweise.

```
#cat /proc/dma  
2: floppy  
4: cascade
```

Unsere Netzwerkkarte, die wir durch ihre Ressourcen begleitet haben, verwendet keine DMA-Kanäle.

Wenn in irgendeiner Form Peripheriegeräte neu angeschlossen werden, kann man aufgrund dieser Angaben und der Angaben in den mit der Hardware mitgelieferten Manuals die Hardware so konfigurieren, dass keine Konflikte mit IRQ, I/O-Port u.s.w. zustande kommen bzw. aufgrund dieser Informationen darauf reagiert werden kann.

Wenn in einem PC eine ISA-Bus-Karte eingebaut werden soll, funktioniert das PnP nicht mehr. Linux stellt uns deshalb Tools zur Verfügung, mit denen man auch diese Karten konfigurieren und in das System integrieren kann.

Das Tool, das wir dazu verwenden können, ist **isapnp**. Das Konfigurationsfile dazu heißt **/etc/isapnp.conf**. Wir sollten die Ausgabe des Kommandos **pnpdump** in die Datei **/etc/isapnp.conf** umleiten.

```
#pnpdump > /etc/isapnp.conf
```

Nun muss noch die Datei **isapnp.conf** so editiert werden, dass die notwendigen Einträge einkommentiert, also das führende #-Zeichen weggelöscht wird.

Danach führen wir den folgenden Befehl aus:

```
#isapnp /etc/isapnp.conf
```

Und erhalten z. B. folgende Ausgabe:

```
Board 1 has Identity e3 33 ff a2 ff ff ff b2 11: TER1544 Serial No: 99
```

Wenn wir annehmen, es handelt sich dabei um eine Soundblaster ViBRA 16C PnP-Karte, also eine Soundkarte, können wir mit **insmod** oder **modprobe** das jeweilige Modul in den Kernel integrieren. Vorausgesetzt ist hier, dass die notwendige

Kernelunterstützung bereits vorliegt. Wenn nicht, muss man einen neuen Kernel mit dieser Unterstützung zuerst bauen.

```
#insmod sound  
oder  
#modprobe sound
```

Wenn bisher alles korrekt abgelaufen ist, können wir mit dem folgenden Befehl den Status der Karte abfragen:

```
#cat /dev/sndstat  
Sound Driver:3.5.4-960630 (Tue Dec 8 23:07:21 MET 1998  
root,  
Linux dedi 2.0.35 #11 Thu Dec 3 22:53:28 MET 1998 i586  
unknown)  
Kernel: Linux dedi 2.0.35 #13 Tue Dec 8 23:18:35 MET  
1998 i586  
Config options: 0  
Installed drivers:  
Type 1: OPL-2/OPL-3 FM  
Type 2: Sound Blaster  
Type 7: SB MPU-401  
Card config:  
Sound Blaster at 0x220 irq 5 drq 1,5  
SB MPU-401 at 0x330 irq 5 drq 0  
OPL-2/OPL-3 FM at 0x388 drq 0  
Audio devices:  
0: Sound Blaster 16 (4.13)  
Synth devices:  
0: Yamaha OPL-3  
Midi devices:  
0: Sound Blaster 16  
Timers:  
0: System clock  
Mixers:  
0: Sound Blaster
```

Nun können wir auf einfachste Weise auch die Funktionstüchtigkeit der Karte überprüfen, indem wir einfach eine Musikdatei auf das Gerät kopieren:

```
#cat musik.au > /dev/audio
```

Die Aufnahmefunktion kann einfach mit

```
#cat < /dev/audio > meinemusik.au
```

überprüft werden. Wir sehen also, dass wir auch ohne einer eigenen Clientsoftware, Musik aufnehmen und wiedergeben können. Sicherlich, dies ist nicht der komfortabelste Weg, aber es ginge.

Wenn wir allerdings nicht nur ISA PnP-Karten im System haben, sondern auch PCI-Karten, dann wird mit sehr großer Wahrscheinlichkeit das System diese selbst erkennen. Wir können aber den Befehl **lspci** dazu verwenden, um uns ausgeben zu lassen, welche PCI-Geräte in unserem System vorhanden sind.

```
#lspci
```

```
00:00.0 Host bridge: Intel Corp. 82815 815 Chipset Host Bridge and Memory Controller Hub (rev 02)
```

```
00:01.0 PCI bridge: Intel Corp. 82815 815 Chipset AGP Bridge (rev 02)
```

```
00:1e.0 PCI bridge: Intel Corp. 82801AA PCI Bridge (rev 02)
```

```
00:1f.0 ISA bridge: Intel Corp. 82801AA ISA Bridge (LPC) (rev 02)
```

```
00:1f.1 IDE interface: Intel Corp. 82801AA IDE (rev 02)
```

```
00:1f.2 USB Controller: Intel Corp. 82801AA USB (rev 02)
```

```
00:1f.5 Multimedia audio controller: Intel Corp. 82801AA AC'97 Audio (rev 02)
```

```
01:00.0 VGA compatible controller: nVidia Corporation NV5M64 [RIVA TNT2 Model 64/Model 64 Pro] (rev 15)
```

```
02:08.0 Ethernet controller: Accton Technology Corporation SMC2-1211TX (rev 10)
```

In dieser Ausgabe finden wir an letzter Stelle unsere SMC-Netzwerkkarte wieder.

**Tabelle 126: Optionen für lspci**

<b>Optionen</b>	<b>Bedeutung</b>
-v	Gibt detaillierte Informationen aus.
-vv	Gibt alle Informationen aus, die ein PCI-Gerät fähig ist auszugeben.
-n	Gibt den Hersteller und den Gerätecode als Nummern aus.
-x	Für debugging-Zwecke werden die ersten 64 Bytes des Konfigurationsraumes ausgegeben.



-xxx	Wie -x, allerdings wird der gesamte Konfigurationsraum ausgegeben.
-t	Ausgabe in Baumform.
-m	Ausgabe in maschinenlesbarer Form.

Wir können aber auch wieder unser `/proc`-Verzeichnis bemühen, um an Informationen über PCI-Geräte zu kommen. Wir geben wieder auf der Kommandozeile Folgendes ein:

```
#cat /proc/pci
PCI devices found:
  Bus 0, device 0, function 0:
    Host bridge: Intel Corp. 82815 815 Chipset Host Bridge and Memory
    Controller Hub (rev 2).
      Prefetchable 32 bit memory at 0x48000000 [0x4bffffff].
    . . .
    . . .
  Bus 0, device 31, function 5:
    Multimedia audio controller: Intel Corp. 82801AA AC'97 Audio (rev
2).
      IRQ 5.
      I/O at 0x3000 [0x30ff].
      I/O at 0x3400 [0x343f].
  Bus 1, device 0, function 0:
    VGA compatible controller: nVidia Corporation RIVA TNT2 Model 64
(rev 21).
      IRQ 5.
      Master Capable. Latency=64. Min Gnt=5.Max Lat=1.
      Non-prefetchable 32 bit memory at 0x41000000 [0x41ffffff].
      Prefetchable 32 bit memory at 0x46000000 [0x47ffffff].
  Bus 2, device 8, function 0:
    Ethernet controller: Accton Technology Corporation SMC2-1211TX
(rev 16).
      IRQ 10.
      Master Capable. Latency=66. Min Gnt=32.Max Lat=64.
      I/O at 0x2000 [0x20ff].
      Non-prefetchable 32 bit memory at 0x40400000 [0x404000ff].
```

Dieses Ergebnis entspricht im Großen und Ganzen der Ausgabe des `lspci`-Befehls mit der Option `-vv`.

#### *Hinweis*

Die meisten Befehle, die systemspezifische Daten ausgeben, wie z. B. `lspci`, greifen eigentlich auf das `/proc`-Verzeichnis zu und offerieren die Ausgabe nur in einer „schöner“ aufbereiteten

Form. Wenn es einmal doch erforderlich wäre, ein PCI-Gerät manuell zu konfigurieren, kann man das Programm **setpci** verwenden.

Oft findet man in Workstations und insbesondere bei Servern **SCSI**-Geräte vor. SCSI ist ein standardisiertes Interface für vielfältigste Geräte wie z. B. Festplatten, Scanner und Bandlaufwerke. Die IDE-ATA-Festplatten haben in letzter Zeit sehr viel an Nachteilen gegenüber SCSI-Festplatten aufgeholt. Dennoch zeichnen sich SCSI-Festplatten, generell SCSI-Geräte, durch höhere Flexibilität und Skalierbarkeit sowie durch höheren Datendurchsatz aus.

SCSI definiert einen Bus-Standard, an dem mehrere unterschiedliche Geräte angeschlossen werden können. Dieser Bus muss aber dann auch richtig terminiert werden. Die Geräte werden in Serie geschaltet, und eines davon ist der sogenannte SCSI-Kontroller.

Dieser SCSI-Kontroller ist im Normalfall eine PCI oder ISA-Einsteckkarte für den PC. Es kann allerdings auch vorkommen, dass der SCSI-Kontroller bereits auf dem Motherboard untergebracht ist. Jeder SCSI-Kontroller besitzt ein SCSI-BIOS. Bei Kontrollern, die auf dem Motherboard untergebracht sind, ist dieses über das „normale“ BIOS zu erreichen und zu konfigurieren. Bei einigen ist das SCSI-BIOS durch die Tastenkombination **STRG-A** zu erreichen, und manche, vor allem sehr billige, lassen überhaupt keinen Eingriff in ihr BIOS zu. Im BIOS werden die Parameter, mit denen der BUS arbeiten soll, gesetzt. So wird hier definiert, welche SCSI-ID der Kontroller selbst hat. Mit welchem Datendurchsatz gearbeitet werden soll, wie mit Wechselmedien umgegangen werden soll sowie welche SCSI-ID bootbar ist. Im Computer BIOS muss eventuell bei der Bootreihenfolge noch eingestellt werden, dass SCSI vor IDE kommt, denn ansonsten wird immer versucht, von den IDE-Platten zu starten, da dies die Defaulteinstellung darstellt.

Jedes Gerät bekommt bzw. muss, bevor es an den Bus angeschaltet wird, eine am Bus eindeutige SCSI-ID (Identifikationsnummer) eingestellt bekommen. Die Einstellung dieser Nummer erfolgt normalerweise über Jumper oder Microschalter, je nach verwendetem Typ.

**Tabelle 127: SCSI Typen**

<b>Typ</b>	<b>Beschreibung</b>
SCSI-1	Das Original: 8-bit, 5MBps Centronics 50 Pin.
SCSI-2	8-bit, 5MBps, Micro-D 50-pin.
Wide-SCSI	16-bit, 10MBps, Micro-D 68-pin.
Fast-SCSI	8-bit, 10MBps, Micro-D 50-pin.
Fast Wide SCSI	16-bit, 20-MBps, Micro-D 68-pin.
Ultra SCSI	8-bit, 20MBps, Micro-D 50-pin.
Ultra Wide SCSI (SCSI-3)	16-bit, 40MBps.
Ultra2	8-bit, 40MBps.
Wide Ultra2	16-bit, 80MBps.
Ultra160-SCSI	16-bit, 160MBps..
Ultra3 SCSI	16-bit, 160MBps.
Ultra320 SCSI	16-bit, 320MBps.
Ultra640 SCSI	16-bit, 640MBps.

Die angesprochene SCSI-ID ergibt sich in einfacher Form aus der Bus-Breite. Also bei 8-Bit können 8 Geräte inklusive Kontroller adressiert werden. Es stehen uns also noch 7 Nummern für weitere Geräte zur Verfügung. Bei 16 Bit kommen wir also auf maximal 15 zusätzlichen Geräte.

Nun gibt es hin und wieder den Fall, insbesondere bei Server, dass an diesem ein **RAID** angeschlossen werden soll. Also ein Speichermedium, das aus mehreren SCSI-Festplatten besteht, aber logisch zu einer zusammengefasst wird. Demzufolge erscheint dieses Gerät am SCSI-Bus mit nur einer SCSI-ID. Damit der Kontroller die einzelnen, hinter dieser Adresse stehenden Platten unterscheiden kann, behilft man sich mit der sogenannten **LUN** (Logical Unit Number). Bei einzelnen Platten oder

Bandlaufwerken wird keine LUN verwendet bzw. auf Null gesetzt.

Wir haben schon angesprochen, dass der SCSI-Bus terminiert werden muss, und zwar auf beiden Enden. Wenn nun der SCSI-Kontroller in der Mitte des Busses sitzt, benötigen wir links und rechts davon eine Terminierung. Die Terminierung dient dazu, elektrische Reflexionen und Störungen, die auf der Leitung, also dem Bus, entstehen, zu kompensieren.

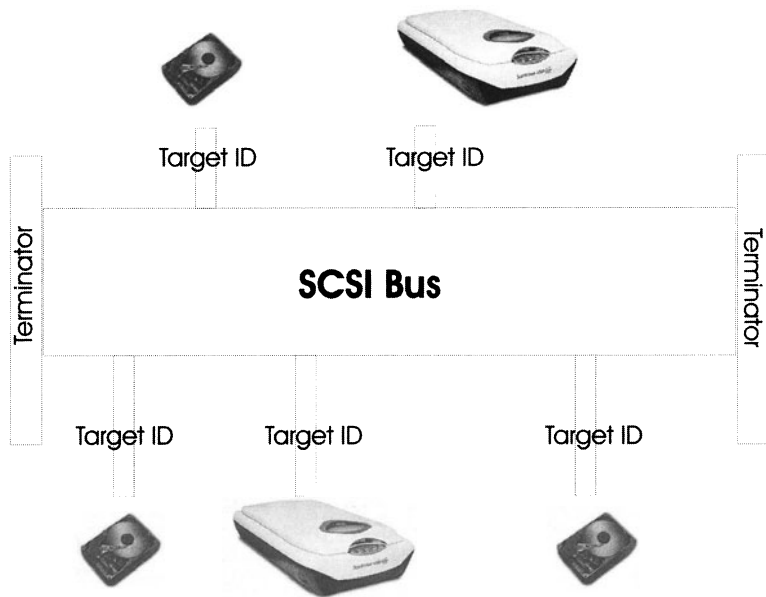


Abbildung 17: Der SCSI Bus

Wenn ein SCSI-Kontroller im PC eingebaut ist und nur extern SCSI-Geräte angeschaltet sind, muss der Kontroller selbst die Terminierung übernehmen. Die Terminierung erfolgt durch Abschlusswiderstände, die entweder durch Jumper, Schalter oder durch Steckwiderstände realisiert werden können. Informationen über SCSI-Geräte bzw den SCSI-Bus erhalten wir mit dem Befehl **scsi\_info**. Der Aufruf dieses Programmes erfolgt einfach so:

```
#scsi_info /dev/scd0
```

Damit erhalten wir Informationen über das SCSI-CD-ROM-Laufwerk. Wir können aber genauso gut unser altbekanntes Verzeichnis `/proc` verwenden und in diesem das Verzeichnis **scsi** inspi-

zieren. So erhalten wir z. B. bei einem Rechner mit CD-Brenner, da dieser mittels SCSI-DIE-Emulation verwaltet wird, folgenden Eintrag:

```
#cat /proc/scsi/ide-scsi/0  
SCSI host adapter emulation for IDE ATAPI devices
```

In modernen Rechnern schon als Standard vorhanden, ist der sogenannte USB-Bus (Universal Serial Bus). Mit diesem Bus kann man nicht nur eine serielle Verkabelung realisieren, sondern auch eine sogenannte Sternverkabelung.

Damit der USB-Bus und damit die USB-Geräte verwendet werden können, muss natürlich der Kernel darauf eingestellt sein. Wenn keine Kernelunterstützung vorhanden ist, muss man den Kernel neu konfigurieren und kompilieren.

Normalerweise wird die USB-Unterstützung als Modul realisiert. Bei allen neuen Distributionen wird USB bereits standardmäßig unterstützt.

Die Module, die die USB-Unterstützung realisieren, sind **usbcore** und **usb-ubci** oder gleichbedeutend **usb-obci**. Wir wissen, dass diese Module die Endung **.o** haben und im Verzeichnis **/lib/modules/Kernel-Version/\*** zu finden sind. In unserem Fall sind die Module in **/lib/modules-2.4.20-8/kernel/drivers/usb** zu finden. Zum Aktivieren der Module geben wir folgende Befehle ein:

```
#insmod usbcore  
#insmod usb-uhci  
oder auch  
#insmod usb-ohci
```

Mit dem Befehl **dmesg** kann man dann nach dem Starten überprüfen, ob der USB-Kontroller erkannt ist und wieviele Ports vorhanden sind, auch wenn keine Geräte angeschlossen wurden. Wir dürfen allerdings nicht vergessen, dass jedes Gerät in der Datei **/etc/modules.conf** mit einem Eintrag vertreten sein muss. Für ein USB-Diskettenlaufwerk kann dieser Eintrag z. B. folgendermaßen aussehen.

```
alias scsi_hostadapter usb-storage
```

Da aber das auf Vorrat-Laden von allen möglichen Geräteunterstützungen für den USB-Bus nicht sinnvoll ist, gibt es ab der

Kernel-Version 2.4 einen Lade-Mechanismus für PC-Cards, USB, ... .

Wir können diesen Mechanismus mit Folgendem aktivieren:

```
#echo "/sbin/hotplug" > /proc/sys/kernel/hotplug
```

Die zu dem Programm **hotplug** gehörenden Dateien kann man in dem Verzeichnis **/etc/hotplug/** finden. Das Programm hotplug inspiziert das Verzeichnis **/etc/hotplug/usb/\*** und entscheidet, welcher Treiber für welches Gerät nachzuladen ist.

Die Datei **/proc/bus/usb/devices** listet alle angeschlossenen USB-Geräte auf. Aus der Datei **/proc/bus/usb/drivers** kann man alle geladenen Kernel-Module für USB auflisten lassen.

Man kann auch den Befehl **usbmodules** benutzen, um Informationen über Kernel-Modul-Treiber für angeschlossene USB-Geräte zu bekommen.

Eine weitere Möglichkeit (Kernel 2.2.x) um USB-Geräteinformationen zu verwalten, ist die Kombination von **usbd** und der Datei **/etc/usbmgr/usbmgr.conf**. Diese enthält pro Gerät einen Eintrag.

Der enorme Erfolg von Apple und Windows wurde unter anderem von der grafischen Benutzeroberfläche und grafisch orientierten Programmen bestimmt. Auch unter Linux gibt es die Möglichkeit, eine grafische Oberfläche zu nutzen.

Wir haben zwar bereits unter einer grafischen Oberfläche gearbeitet, werden nun aber einen Schritt zurück machen und diese Oberfläche ausschalten, indem wir den Computer neu installieren und X11 nicht mitinstallieren.

### Übung

Da die Gelegenheit gleich passt, sollten Sie nun versuchen, unsere Red Hat-Version ohne grafische Oberfläche zu installieren.

## 9.1

### Der Grafikzauber

Wir wollen unserem System eine grafische Oberfläche verpassen. Linux verwendet das sogenannte **X-Window-System**. Die Treiber für die verschiedensten Grafikkarten werden von dem XFree86-Team ([www.xfree86.org](http://www.xfree86.org)) entwickelt. Allerdings gibt es immer mehr Grafikkartenhersteller, die eigens entwickelte Treiber anbieten und damit auch die Leistungsfähigkeit der Karte wesentlich besser ausreizen können. Einer dieser Hersteller ist z. B. nVidia. XFree86 ist die freie Implementation des X-Window-Systems, das auf allen Unix-Systemen läuft. Die 86 im Namen kommt aus der Geschichte, als das System nur für x86 Systeme entwickelt wurde, mittlerweile gibt es Portierungen auf verschiedene Architekturen. Damit man unter Linux mit einem grafischen System arbeiten kann, benötigt man einen sogenannten X-Server, der das Bindeglied zur Grafikkarte darstellt.

Wir wollen im Folgenden die neue Version 4.3.0 von XFree86 installieren.

Wie bei allen Geräten, die man bei Linux in das System integrieren möchte, ist es notwendig, detaillierte Informationen über die jeweilige Hardware zu besitzen. Bei der Vielzahl von am Markt angebotenen Grafikkarten und Herstellern kann es mühsam werden, den Hersteller bzw. die Grafikkarte des eigenen PC's zu

bestimmen. Ganz schlimm wird es, wenn der Grafikchip auf dem Motherboard untergebracht ist. Man braucht aber nicht unbedingt den richtigen Hersteller der Karte, sondern vielmehr den richtigen Grafikchip, der auf der Karte verwendet wird. Oft kommt man nicht drumherum, den Rechner zu zerlegen und einen direkten Blick auf die Grafikkarte zu werfen.

Das XFree86-Team unterstützt uns bei der Suche nach der richtigen Grafikkarte, indem sie sie eigentlich als Black Box behandelt. Wenn wir auf dem Downloadserver der aktuellen Xfree86-Version sind, finden wir dort eine Datei, die **Xinstall.sh** heißt. Mit dieser Datei kann man in Erfahrung bringen, für welche Architektur mit welchen Laufzeitbibliotheken man XFree86 herunterladen muss, und sie beinhaltet alle unterstützten Grafikkarten.

Wir geben nach erfolgreichem Download Folgendes ein:

```
#sh Xinstall.sh -check
Checking which OS you're running...
uname reports 'Linux' version '2.4.20-8', architecture 'i686'.
Object format is 'ELF'.  libc version is '6.3.2' (6.3).
```

```
Binary distribution name is 'Linux-ix86-glibc23'
```

If you don't find a binary distribution with this name, then binaries for your platform are not available from XFree86.org

Der wichtige Punkt dabei ist die Zeile:

```
Binary distribution name is 'Linux-ix86-glibc23'
```

Damit wissen wir, aus welchem Binaries-Verzeichnis wir die benötigten XFree86-Programme herunterladen müssen.

Wir wechseln also in das angegebene Verzeichnis und laden uns die folgenden notwendigen Programme herunter:

- |    |             |                             |
|----|-------------|-----------------------------|
| 1. | Xinstall.sh | Das Installationsskript     |
| 2. | extract     | Für das Entpacken der tgz's |
| 3. | Xbin.tgz    | X-Clients/Utilities         |
| 4. | Xlib.tgz    | Laufzeitbibliotheken        |
| 5. | Xman.tgz    | Manual Pages                |
| 6. | Xdoc.tgz    | Dokumentation               |
| 7. | Xfnts.tgz   | Font-Basis                  |



8.	Xfenc.tgz	Font-Kodierungsdaten
9.	Xetc.tgz	Konfigurationsdateien
10.	Xvar.tgz	Laufzeitdaten
11.	Xxserv.tgz	X-Server
12.	Xmod.tgz	X-Server-Module

Diese 12 Programme sind unbedingt notwendig. Die weiteren 11 Programme stellen nur zusätzliche Optionen dar, die zur Funktion nicht unbedingt notwendig sind. Wir wollen diese aber dennoch herunterladen.

1.	Xsrv.tgz	Font-Server
2.	Xnest.tgz	Eingebettete X-Server
3.	Xprog.tgz	Entwicklerunterstützung
4.	Xprt.tgz	X-Printer-Server
5.	Xvfb.tgz	Framebuffer-Unterstützung
6.	Xf100.tgz	100dpi Fonts
7.	Xfcyr.tgz	Cyrillic Fonts
8.	Xfscl.tgz	Skalierbare Fonts
9.	Xhtml.tgz	HTML-Version der Doku
10.	Xps.tgz	PostScript-Version der Doku
11.	Xjdoc.tgz	Japanische Doku

Vom XFree86-Team wird verbindlich empfohlen, das XFree-System nur Mithilfe des **Xinstall.sh**-Skripts zu installieren. Wir wollen dieser Empfehlung natürlich Folge leisten und wechseln in das Verzeichnis, in das wir unsere Dateien vom XFree86-Server gespeichert haben. Nehmen wir an in **/root/xfree**.

```
#sh Xinstall.sh
```

Sie werden nun vom Installationsskript einige Fragen präsentiert bekommen, die man entweder mit Ja [y] oder Nein [n] beantworten kann.

Wenn man schon eine XFree Installation auf dem Rechner hat, wird man gewarnt bzw. gefragt, ob man damit einverstanden ist, wenn diese überschrieben wird. Hier sollte man in diesem Fall kurz inne halten und überlegen, was das bedeuten kann, wenn

bei der Installation etwas schief geht. In unserem Fall haben wir ja keine funktionierende XFree86-Installation, und damit können wir dies mit ruhigen Gewissen bejahen.

Die Konfigurationsdaten werden während der Installation in das Verzeichnis **/etc/X11** geschrieben. Bei XFree86-Versionen bis 3.9.18 stehen diese in **/usr/X11R6/lib/X11**. Wenn man über eine bestehende Installation installiert, erkennt das Installationskript die unterschiedlichen Versionen und setzt selbständig einen Link. Durch eine Abfrage kann man auch mit `n` bestimmen, dass die Konfigurationsdateien nicht in **/etc/X11** kommen soll.

Wenn wir optionale Komponenten vom XFree86-System heruntergeladen haben, werden wir pro Komponente gefragt, ob wir diese installieren wollen oder nicht.

Nun sind wir eigentlich mit dem Hauptpfad der Installation schon fertig. Jetzt wird noch **ldconfig** aufgerufen, damit die neu installierten Libraries auf dem System auch verfügbar gemacht werden. Anschließend werden noch die neuen Fonts upgedated.

Abschließend wird noch gefragt, ob man einen Link auf **rstartd** legen will, was bei den meisten Systemen nicht notwendig ist, und auch wir wollen den **remote start client** nicht haben. Der remote start client verwendet **rsb** (remote shell), welche eigentlich aus Sicherheitsgründen nicht verwendet werden sollte.

Damit ist die Installation abgeschlossen. Wir benötigen noch eine Grundkonfigurationsdatei für unseren X-Server. Es gibt derzeit drei Wege, wie man zu so einer Basiskonfiguration kommt. Der eine Weg ist das Tool **xf86config**. Der andere Weg ist das **xf86cfg**-Tool und der dritte – den wollen wir beschreiten – ist der neue Weg und ist eine Option für **XFree86**.

```
#XFree86 -configure
```

Dieser Befehl erzeugt für uns eine Grundkonfigurationsdatei **XF86Config**. Das Format der Datei hat sich seit der Version 3 geändert, und auch der Pfad hat sich nun, wie bereits oben besprochen, auf **/etc/X11** geändert.

Wir können unseren X-Server nun ausprobieren, indem wir folgenden Befehl eingeben:

```
#XFree86 -xf86config /root/xfree/XF86Config.new
```

Das Programm **XFree86** ist der eigentliche X-Server für unser System, also für unsere Grafikkarte. Wenn wir zufrieden sind mit der Funktion unseres X-Server, können wir alle alten Dateien **XF86\_\*** und/oder **XF98\_\*** im Verzeichnis **/usr/X11R6/bin/** löschen. Des Weiteren können wir nun das Konfigurationsfile **XF86Config.new** nach **/etc/X11/XF86Config** kopieren.

Den X-Server können wir nun manuell nach jedem Starten mit dem Befehl **startx** aktivieren. Der Computer schaltet dann in den grafischen Modus.

Neben dieser neuen automatischen Methode gibt es, wie bereits gesagt, noch weitere Möglichkeiten, den X-Server zu konfigurieren. **xf86config** ist textbasierend, die Tools **XF86Setup** und **xf86cfg** sind bereits grafisch basierend, die auf einem VGA X-Server aufsetzen. Dieser Modus wird von allen Grafikkarten unterstützt. Des Weiteren kann man mit diesen Tools auch nachträglich Änderungen in der Konfigurationsdatei komfortabel durchführen.

Mit dem Tool **xvidtune** (video mode tuner) können die Einstellungen (mode-Zeilen) der **XF86Config**-Datei direkt bearbeitet werden und auch als solche Mode-Zeile direkt auf der Konsole ausgegeben werden. Dieses Tool ist ein Clientinterface in die X-Server Video Modus Extension.

#### *Achtung*

Es ist bei diesem Tool aber Vorsicht geboten, denn falsche Einstellungen im Videomodus der Grafikkarte kann diese oder den Monitor beschädigen. Eine dementsprechende Warnmeldung wird beim Start des Programmes ebenfalls angezeigt und sollte ernst genommen werden.

Es stehen die Buttons **left**, **right**, **up**, **down**, **wider**, **narrower**, **shorter** und **taler** für das Anpassen des Displays zur Verfügung. Mit **show** kann man sich die Modezeile für die **XF86Config**-Datei auf der Konsole ausgeben lassen. Mit **next** kann man zum nächsten Videomodus weiterschalten.

Nun wollen wir unseren X-Server mit dem **xf86cfg**-Tool konfigurieren. Wir wollen allerdings nur die Auflösung und die Tastatur damit einstellen.

```
#xf86cfg
```

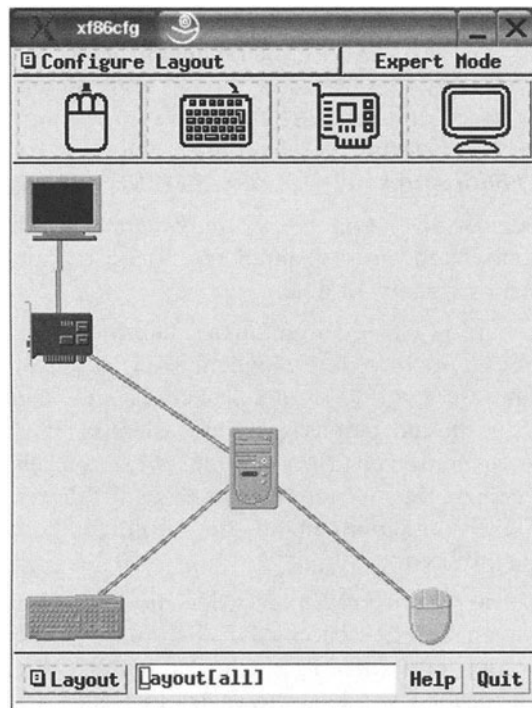


Abbildung 18a: xf86cfg Startbildschirm



Abbildung 18b: xf86cfg Startbildschirm-Bedienelemente

Wir gehen mit der Maus, wenn diese nicht funktionieren sollte, mit den angegebenen Tastenkombinationen, auf den Menüpunkt

**Configure Layout** und wählen den Eintrag für **Screen**. Mit einem rechten Mausklick auf den nun erschienenen Monitor können wir den Punkt **configure** auswählen. Hier können wir für unterschiedliche Farbtiefen unterschiedliche Auflösungen zuteilen. Wir wählen 16 und 1024x768.

Unter Configure Layout klicken wir wieder mit der rechten Maustaste auf die Tastatur und wählen configure. Hier stellen wir Generic 104 Tasten und German sowie eliminate dead keys ein. Wir bestätigen alles mit **Apply Changes** und **OK**.

Nun steigen wir aus dem Tool mit **QUIT** wieder aus. Doch zuvor erhalten wir die Aufforderung, die Datei zu speichern, und bekommen einen Vorschlag. Wenn wir uns sicher sind, dass diese Konfiguration akkurat ist, können wir auch gleich unter `/etc/X11/XF86Config` speichern. Probieren wir es. Auch die zweite Aufforderung ändern wir um in den Pfad `/etc/X11`.

Wieder auf dem Kommand Prompt angelangt, starten wir den X-Server durch Eingabe von `startx`. Wir gelangen in die grafische Oberfläche, allerdings in die Auflösung 640x480. Wir wollen aber in unsere konfigurierte Auflösung schalten und zwar zu 1024x768. Wir können dies mit der Tastenkombination **STRG+ALT** und die **+** Taste auf dem Ziffernblock erreichen. Mit dieser Tastenkombination kann man durch alle konfigurierten Auflösungen wechseln.

Eine weitere Möglichkeit, den X-Server zu konfigurieren ist das textbasierte Tool `xf86config`.

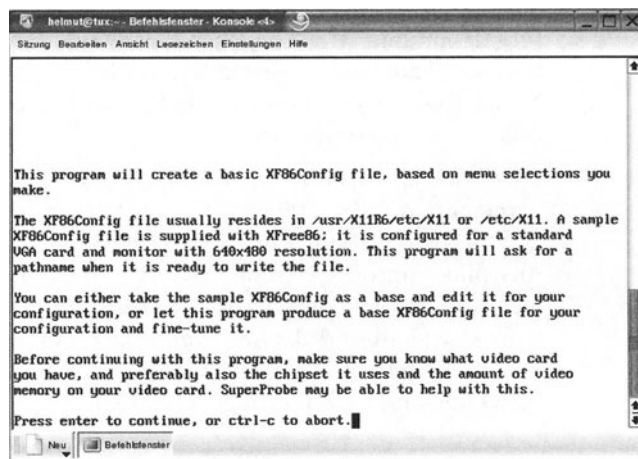


Abbildung 19: Startbildschirm vom `xf86config`-Tool

Wir werden mit mehreren Bildschirmen und Fragen konfrontiert, die wir beantworten und am Ende eine XF86Config-Datei erhalten.

Zuerst müssen wir die Art der Maus und damit das verwendete Protocol für das Pointer-Device festlegen.

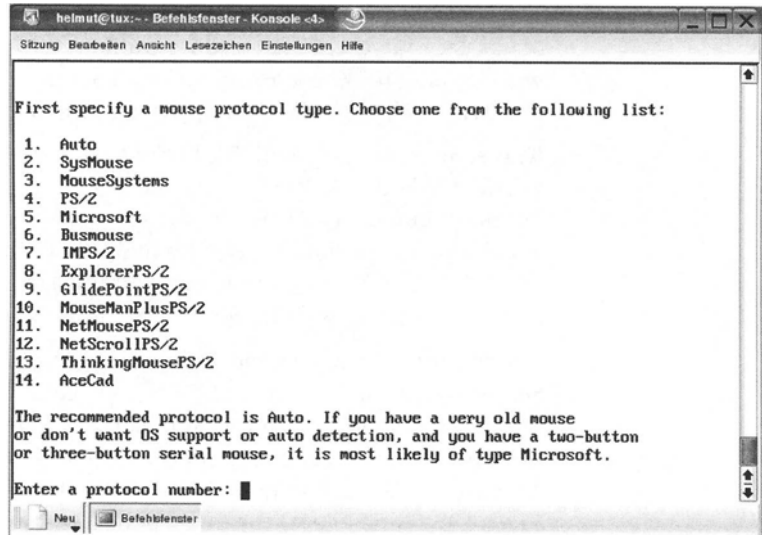


Abbildung 20: Mausauswahl

Nach dieser Auswahl können wir festlegen, ob wir eine Drei-Tasten-Emulation haben wollen oder nicht. Da unter X-Windows und damit unter UNIX/Linux bei der Maus schon immer drei Tasten unterstützt und verwendet werden, ist diese Emulation für Mäuse mit nur zwei Tasten, die bis vor kurzem bei PC-Systemen üblich waren, vorgesehen.

Wenn wir also eine Drei-Tasten-Maus oder Wheel-Maus besitzen, verneinen wir diese Frage. Als nächstes wird das Mausinterface (seriell, usb, ...) erfragt. Wenn wir unsere Maus auf dem PS/2 Anschluss unseres Computers angeschlossen haben, dann verwenden wir das Device **/dev/psaux**. Anschließend wird nach dem Keyboard und darauffolgend nach der Sprache und Sprachbelegung gefragt. Danach werden die Informationen für den verwendeten Monitor erfragt.

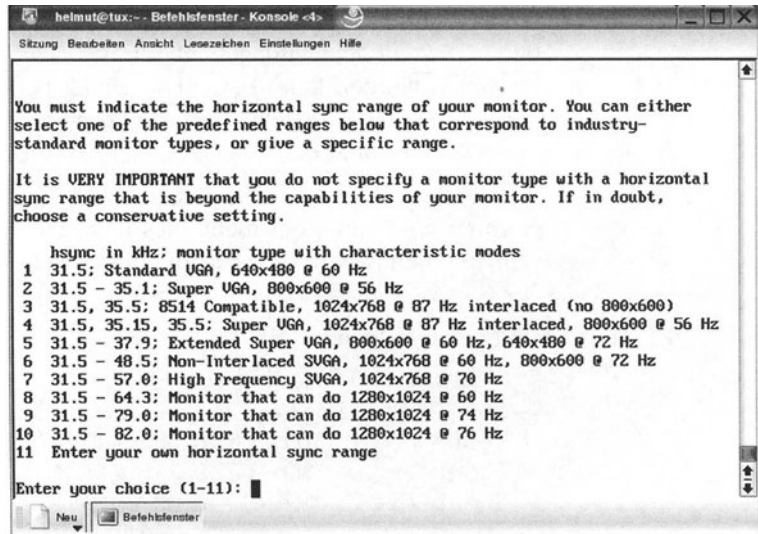


Abbildung 21: Monitorauswahl

Für die Grafikkarte können wir einen Blick in eine Datenbank werfen, ob unsere eingebaute Grafikkarte unterstützt wird.

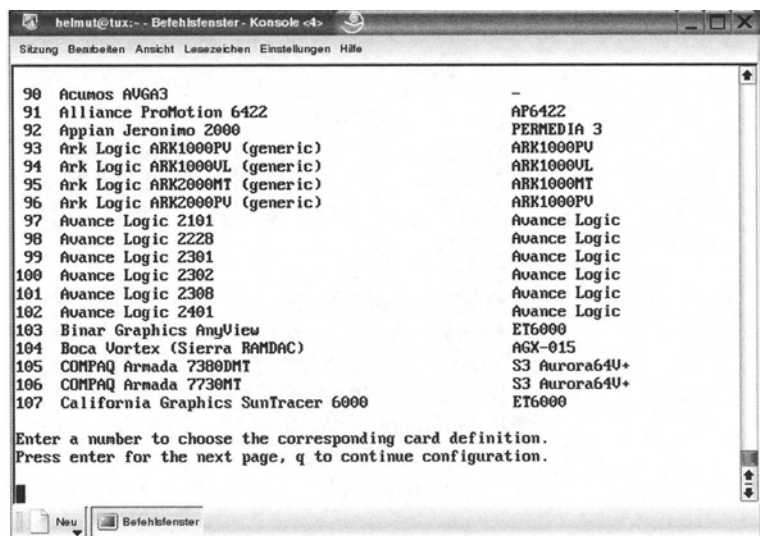


Abbildung 22: Grafikkarten Datenbank

Jetzt werden das Video-Ram und die default-Farbtiefe eingegeben. Letztendlich wird gefragt, ob /etc/X11/XF86Config überschrieben werden kann bzw. darf. Damit haben wir wieder eine XF86Config-Datei erstellt, und wir können deren Funktion mit startx kontrollieren.

Wir wollen, dass standardmäßig unsere gewünschte Auflösung gestartet wird. Um aber nicht alles noch einmal machen zu müssen, nehmen wir uns das Konfigurationsfile gleich mit dem vi vor.

Wir geben Folgendes ein:

```
#vi /etc/X11/XF86Config
Section "ServerLayout"
    Identifier      "XFree86 Configured"
    Screen          0  "Screen0"  0 0
    InputDevice     "Mouse0"  "CorePointer"
    InputDevice     "Keyboard0" "CoreKeyboard"
EndSection

Section "Files"
    RgbPath          "/usr/X11R6/lib/X11/rgb"
    ModulePath        "/usr/X11R6/lib/modules"
    . . .
    . . .
    #Option          "FPDither"          # [<bool>]
    #Option          "CrtcNumber"        # <i>
    Identifier       "Card0"
    Driver            "nv"
    VendorName        "nVidia Corporation"
    BoardName         "NV5M64 [RIVA TNT2 Model 64/Model 64 Pro]"
    BusID             "PCI:1:0:0"
EndSection

Section "Screen"
    Identifier       "Screen0"
    Device           "Card0"
    Monitor          "Monitor0"
    DefaultDepth      16
    SubSection       "Display"
        Depth        1
    EndSubSection
    SubSection       "Display"
        Depth        4
    EndSubSection
    SubSection       "Display"
```



```

        Depth      8
    EndSubSection
    SubSection "Display"
        Depth      15
    EndSubSection
    SubSection "Display"
        Depth      16

        Modes "1024x768" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      24
    EndSubSection
EndSection

```

Wir sehen, dass diese Datei in Sektionen unterteilt ist. Die uns interessierende Sektion ist die **Screen**-Sektion und hierin die **SubSection "Display"** für unsere Farbtiefe von 16 Bit. (Fett unterlegt.) Hier sehen wir, dass die Einträge für die Auflösung hintereinander stehen. Damit wir nun die 1024x768 zuerst erhalten, also standardmäßig in dieser Auflösung starten, vertauschen wir einfach die Reihenfolge. Aus der fettgedruckten Zeile wird einfach:

```

Modes "1024x768" "640x480"

```

Den Erfolg unserer Änderung wollen wir auch überprüfen. Wir speichern unsere Änderungen ab und beenden den X-Server mit der Tastenkombination **STRG+ALT+BACKSPACE**. Danach geben wir wieder `startx` ein und – „Hurra“ wir haben es geschafft.

Da wir die Version 4 des XFree86 verwenden, haben wir nur einen X-Server, wobei die eigentliche Grafikkarte durch den Eintrag **chipset** im XF86Config-File definiert wird. In der Version 3 ist der X-Server eigentlich ein Link von **X** auf den eigentlichen Server z. B.:

```
X => /usr/X11R6/bin/XF86_SVG
```

Die Version 4 hat also den Vorteil, dass sie alle Server integriert und nur durch den Chipsatz differenziert. Leider ist es aber noch so, dass die Version 4 nicht so viele Karten wie die Version 3 unterstützt, aber dafür die aktuelleren. Die Datei XFree86 ist der Server für die Version 4 und damit das Ziel des Links.

Eine Auswahl an Servernamen für die Version 3 zeigt Tabelle 128.

**Tabelle 128: X-Server**

<b><i>X-Server</i></b>	<b><i>Unterstützter Chipsatz</i></b>
XSVG	VESA Super VGA-Karten.
XS3	S3 basierende-Karten.
XMach64	Für ATI Mach 64.
SP9K	Server für P9000 basierende Karten.
XAGX	AGX basierende Karten.

Sektionen in der XF86Config:

**Tabelle 129: Unterteilung der Konfigurationsdatei**

<b><i>Sektion</i></b>	<b><i>Bedeutung</i></b>
Module	Ladbare Module.
InputDevice	Beschreibung der Eingabegeräte.
Device	Beschreibung der Graphik.
VideoAdaptor	Video Adapter-Beschreibung.
Monitor	Monitor-Beschreibung.
Modes	Video Modi-Beschreibung.
Screen	Bildschirmbeschreibung.
Vendor	Herstellerbeschreibung.

Beim neuen Version 4-Server von XFree86 ist die Sektion ***InputDevice*** der Ersatz für die Sektionen ***Keyboard*** und ***Pointer*** (für die Maus) der Version 3.

Die Sektion ***ServerLayout*** ist der höchste Level. Diese bindet Eingabe- und Ausgabe-Geräte einer Session zusammen.

In der ***File***-Sektion werden einige Pfade, die für die Funktion des Servers notwendig sind, gesetzt. Im ***FontPath*** "***Pfad***" wird ein Suchpfad für Schriften, die der X-Server verwenden kann, in

einer mit Beistrichen getrennten Liste aufgeführt. Die Syntax für diesen Eintrag ist folgendermaßen:

trans/hostname:Port-Nummer

Hierin bedeutet trans den Transporttyp, mit dem die Verbindung mit dem Font-Server erfolgen soll. Hier kann z. B. **unix** stehen, wenn ein unix-socket oder **tcp** für eine TCP/IP-Verbindung verwendet werden soll. Hostname und Port geben den eigentlichen Rechner, auf dem der Font-Server läuft, an und den Port (normalerweise 7100), auf den der Font-Server hört.

Wenn dieser Eintrag nicht vorhanden ist, verwendet der X-Server seine Defaulteinstellungen und sucht Schriften in dem Verzeichnis **/usr/X11R6/lib/X11/fonts/\***. Der **RGBPath "Pfad"** weist den Weg zur RGB-Datenbank. Der Standardpfad dorthin ist **/usr/X11R6/lib/X11/rgb**. Schließlich zeigt der Eintrag **ModulePath "Pfad"** den Weg zu ladbaren X-Server-Modulen.

In der **ServerFlags**-Sektion werden einige globale Optionen gesetzt.

**Tabelle 130: Optionen für den X-Server**

<b>Option</b>	<b>Bedeutung</b>
DontVTSwitch	Verhindert/erlaubt das Umschalten auf virtuelle Konsolen mit der Tastenkombination ALT+STRG+FN (FN = Funktionstaste).
DontZap	Verhindert/erlaubt das Beenden des X-Severs mit der Tastenkombination STRG+ALT+Backspace.
DontZoom	Verhindert/Erlaubt das Umschalten der Auflösungen mit STRG+ALT+Nummerblock-PLUS.
BlankTime	Setzt die Zeit in min., die vergeht, bis der Screensaver schwarz geschaltet wird. Default ist 10 min.

In der **InputDevice**-Sektion werden die Eingabegeräte definiert, z.B:

```

Section "InputDevice"
    Identifier "Name"
    Driver "Inputtreiber"
    Options
    ...
EndSection

```

Die Device-Sektion ist ähnlich aufgebaut und beschreibt die in dem System-Verwendete Grafikkarte näher.

```

Section "Device"
    Identifier "Name"
    Driver "Treiber"
    Einträge
    ...
EndSection

```

**Tabelle 131: Optionen für die Grafikkarte**

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
BusID	Spezifiziert den Steckplatz der Karte (AGP/PCI).
Screen	Dieser Eintrag ist wichtig für Karten oder Systeme, die mehrere Monitore unterstützen sollen.
Chipset	Hier wird der zu verwendende Grafikchipset definiert.
VideoRam	Gibt die Größe des Videospeichers an.
Options	Hier können treiberspezifische und treiberunabhängige Parameter angegeben werden.

Die Monitor-Sektion definiert den verwendeten Monitor:

Section "Monitor"

Identifizier "Name"

Driver "Inputtreiber"

Einträge

...

EndSection

**Tabelle 132: Optionen für den Monitor**

<i><b>Optionen</b></i>	<i><b>Bedeutung</b></i>
HorizSync	Definiert den unterstützten Frequenzbereich für die Horizontalfrequenz an. Default ist 28-33kHz.
VertRefresh	Gibt den unterstützten Frequenzbereich für die vertikale Wiederholfrequenz an. Default ist 43-72Hz.
DisplaySize	Gibt die Weite und Höhe in mm des verwendeten Monitors an.
Gamma	Die Gamma-Korrektur des Monitors. Die Angabe kann entweder als Summe oder als RGB-Wert aufscheinen.
Mode	Beginnt eine multi Video Mode-Definition.
Modline	Spezifiziert eine Zeile mit einem speziellen Video-Modus.

Die Screen-Sektion repräsentiert die Verbindung eines Grafikdevices und einen Monitor. Eine Screen-Sektion wird als aktiv erkannt, wenn sie im ServerLayout referenziert wird. Wenn nichts definiert wird, wird der erste Screen-Sektion-Eintrag verwendet.

```

Section "Screen"
    Identifier "Name"
    Device "Device-ID"
    Monitor "Monitor-ID"
    Einträge

    . . .
    SubSection "Display"
        Einträge
        . . .
        EndSubSection
    . . .
EndSection

```

**Tabelle 133: Optionen für die Screen-Sektion**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
Device	Hier wird die Device-Sektion definiert, die verwendet werden soll.
Monitor	Spezifiziert den Montior, der verwendet werden soll.
VideoAdaptor	Spezifiziert einen zu verwendenen optionalen Videoadapter.
DefaultDepth	Spezifiziert die Default-Farbtiefe.
Option Accel	Aktiviert die von der Karte unterstützte 2D Hardware-Beschleunigung.

**Tabelle 134: Optionen für die Untersektionen**

<b>Option</b>	<b>Bedeutung</b>
Depth	Hier wird die Farbtiefe, die diese Display Subsection verwenden soll, definiert.
Weight	Definiert die RGB Gewichtung, die Verwendung finden soll.
Virtual	Mit Xdim und Ydim kann man einen virtuellen Bildschirm definieren.
Viewport	Setzt die linke obere Ecke des Anfangsschirms x0 y0.
Modes	Gibt eine Liste der zu verwendenden Videomodi an.
Visual	Dieser Eintrag setzt den visuellen default root-Typ. Default ist True-Color für 16 Bit Farbtiefe.
Options	Hier können treiberspezifische und treiberunabhängige Parameter angegeben werden.

Das Konfigurationsfile kann mehrere ServerLayout-Sektionen besitzen. Das ServerLayout kombiniert nun die Ein- und Ausgabegeräte, um eine vollständige Konfiguration zu erhalten.

```

Section ServerLayout
    Identifier Name
    Screen Screen-ID
    ...
    InputDevice Device-ID
    ...
    Optionen
    ...
EndSection

```

**Tabelle 135: Optionen für das Server Layout**

<b>Optionen</b>	<b>Bedeutung</b>
Screen	Referenziert eine ScreenSection. Die einzelnen Bildschirme werden durch eine eindeutige Nummer unterschieden. Erster Screen hat die Nummer Null.
InputDevice	Hier werden normalerweise zwei Einträge erwartet, einer für die Tastatur und einer für die Maus.

Hier sehen wir eine typische ServerLayout-Sektion:  
 Section "ServerLayout"

```

  Identifier      "XFree86 Configured"
  Screen         0  "Screen0"  0  0
  InputDevice     "Mouse0"  "CorePointer"
  InputDevice     "Keyboard0" "CoreKeyboard"

```

EndSection

Bevor wir uns wieder der Art und Weise, wie der X-Server gestartet werden kann, zuwenden, wollen wir noch einen kurzen Blick auf das **WIE** der Schriftenkonfiguration werfen.

Die Schriften eines X-Windows-Systems werden von einem eigenen Serverdienst verwaltet. Dieser Server läuft standardmäßig im Hintergrund. Wenn man neue Schriften installiert, müssen wir dafür sorgen, dass der Font-Server auch von den neuen Schriften erfährt. Der Font-Server weiß aufgrund der File-Sektion in der XF86Config, wo er die Schriften suchen soll und finden kann.

```
FontPath "unix/:-1"
```

Dieser Eintrag weist den Font-Server an, das lokale System (unix als Transportmod, keine Hostangabe und die Port-Nummer -1) zu verwenden.

```
FontPath "tcp/font.powerup.at:7100"
```

Dies weist den Font-Server an, als Transportprotokoll TCP/IP zu verwenden und sich mit dem Server font.powerup.at auf Port 7100 zu verbinden.



Die Schriften sind normalerweise in `/usr/X11R6/lib/X11/fonts` zu finden.

Damit wir ein neues Schriftenverzeichnis und die dafür notwendigen Dateien hinzufügen können, benötigen wir das Kommando **mkfontdir**.

```
#mkfontdir /usr/X11R6/lib/X11/fonts/meineSchriften
```

Mit diesem Kommando wird die Datei **fonts.dir** erstellt. Es muss jedesmal neu ausgeführt werden, wenn ein neuer Schriftsatz hinzugefügt wurde. Eine weitere Datei, die in diesem Verzeichnis vorkommen kann, ist die Datei **fonts.alias**. Diese muss allerdings von Hand erstellt werden und besteht aus zwei Spalten für Alias und Schriftname.

Wenn alles vorbereitet ist, muss man den X-Server veranlassen, das Fonts-Directory und damit die Schriften wieder neu einzulesen. Mit dem **xset** Befehl wird dies erledigt.

```
#xset fp rehash
```

Nun aber zurück zum Starten des X-Servers. Eine weitere Art, den X-Server zu starten, ist das Programm **xinit** (X Windows-System Initializer). Das **xinit**-Programm wird dazu verwendet, den X-Server und ein erstes Client-Programm auf dem System zu starten. X-Windows-Clients sind also Programme, die den X-Server verwenden. Wenn kein spezifisches Client-Programm beim Aufruf von **xinit** angegeben wird, sucht das Programm im HOME-Verzeichnis des betreffenden Benutzers nach einer Datei mit dem Namen **.xinitrc**. Die Datei **.xinitrc** fungiert als Shell-Skript, um Client-Programme zu starten. Wenn eine solche Datei nicht vorhanden ist, startet **xinit** einfach den X-Server und ein einziges Terminal (**xterm**).

```
xterm -geometry +1 +1 -n login -display :0
```

Wenn dieser Client wieder geschlossen wird, terminiert sich auch der X-Server wieder.

Wir können dies in die Datei **.xinitrc** schreiben oder gleich beim Aufruf mit angeben.

```
#xinit -geometry =80x80+10+10 -fn 8x13 -j -fg blue -bg yellow
```

Nachfolgend ist ein Beispiel einer **.xinitrc**, welche zwei Terminal-Fenster, eine Uhr und den **tab Window-Manager** (**twm**) startet.

```

xrdb -load $HOME/.Xresources
xsetroot -solid white &
xclock -g 50x50-0+0 -bw 0 &
xload -g 50x50-50+0 -bw 0 &
xterm -g 80x24+0+0 -bg navy &
xterm -g 80x24+0-0 &
twm

```

Zuerst wird in die X-Server-Rourcedatenbank die **.Xresources** eingelesen und danach die root-Parameter.

Der letzte Eintrag in der Beispiel `.xinitrc`-Datei ist ein Window-Manager. Er stellt das Interface zwischen User und X-Server dar. Der Default-Screen eines X-Servers ist nichts anderes als ein grafischer Bildschirm mit Mauszeiger. Ein Window-Manager liefert den Rahmen, die Titelleiste, Menüs, Icons, ... für die Darstellung auf dem Bildschirm. Der **twm** ist nur einer von vielen Window-Managern, die man mit dem X-Window-System verwenden kann. Das **KDE** (K-Desktop Environment) und **GNOME** sind häufig eingesetzte Window-Manager, welche allerdings weit über die Fähigkeiten eines „normalen“ Window-Managers hinaus gehen und daher meiner Meinung nach besser die Bezeichnung **Desktop-Management-System** verdienen. Die „reinen“ Window-Manager, die häufig anzutreffen sind, heißen:

**Tabelle 136: Window-Manager**

<i><b>Name</b></i>	<i><b>Beschreibung</b></i>
fvwm	High extensible Manger.
icewm	Window-Manager, der mehrere Umgebungen emulieren kann.
amiWM	Amiga workbench Window-Manager.
olwm	Sun's Open Look.
olvwm	Sun's Open Look mit virtuellen Bildschirmen.
AfterStep	Ein NeXt Window-Manager.
Enlightenment	Das ist der Default Window-Manager für die GNOME-Umgebung.

Wenn wir uns für einen Window-Manager entschieden haben, wollen wir eventuell für unsere User ein einheitliches Look and Feel erzeugen, sobald die Benutzer den X-Server mit `startx` aufrufen. Wenn wir mit `startx` aktivieren, wird nicht die Datei `$HOME/.xinitrc` gelesen, sondern die globale **`xinitrc`**-Datei. Ist diese Datei nicht vorhanden, wird im HOME-Verzeichnis des Users nach der Datei `.xinitrc` gesucht. Wenn auch dort keine solche Datei vorhanden ist, wird der X-Server mit den Standardwerten aufgerufen, also ein X-Terminal und sonst nichts. Die Datei `$HOME/.xinitrc` „over ruled“ die globale Datei. Wenn diese vorhanden ist, werden deren Informationen verwendet.

Wir müssen also die globale Datei `/usr/lib/X11/xinit/xinitrc` editieren. Allerdings muss man sich der Einschränkung bewusst sein, dass damit nur der erstmalige Einstieg standardisiert ist. Der User kann dann immer noch mit `.xinitrc` Eingriffe vornehmen. Wir als Administratoren können dies aber auch nutzen, um gewissen Usern eine gewisse Umgebung zur Verfügung zu stellen.

Wir verwenden folgende Datei als globale `xinitrc`.

```
#!/bin/sh
xsetroot -solid darkslateblue
xclock -geometry 96x96-2+2 -bg grey40 -fg black -hl white &
xload -geometry 120x96+2+500 -bg grey40 -fg white -hl darkred -
update 2xclock &
xterm -sb -ls -geom 80x25+2+2 -title "Helmut" &
xterm -sb -ls -geom 80x25-1-1 -title "Helmut 1" &
exec twm
```

Um von der Handarbeit weg zu kommen, wollen wir, dass auch schon der Login-Screen im grafischen Modus vorliegt. Mit **`xdm`** (X Display-Manager ist im Lieferumfang von XFree86 enthalten) kann man dies realisieren. Nach dem Booten präsentiert sich eine grafische Anmeldemaske, die auch sofort gestartet wird, wenn man aus dem X-Server wieder aussteigt.

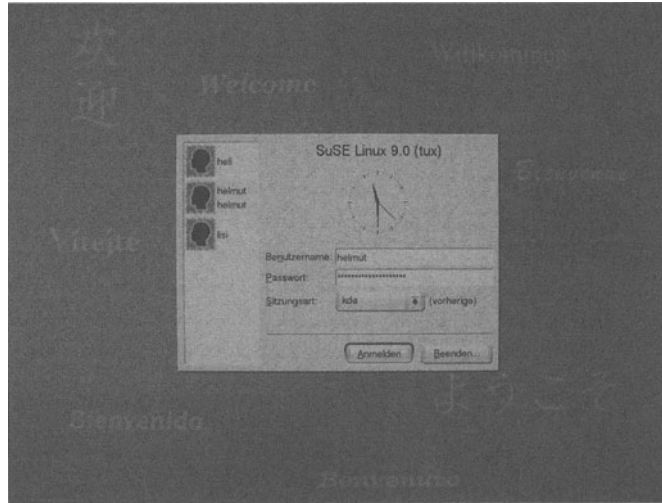


Abbildung 23: Login-Bildschirm - kdm

### Einschub

Die Desktop Systeme KDE und GNOME verwenden jeweils ihre eigenen Display-Manager-Programme, die für den KDE **kdm** und für GNOME **gdm** heißen.

Der xdm wird durch ein System-V Start-Skript im jeweiligen Run-level, z. B. 5 bei Red Hat und SuSE oder 2 bei Debian, gestartet.

Wir erstellen ein Start-Skript in `/etc/init.d` und verlinken es den Konventionen folgend in den entsprechenden Runlevel.

Die Konfigurationsdateien des xdm findet man im Verzeichnis `/etc/X11/xdm/*`. Analog dazu findet man die für den kdm und gdm in `/etc/X11/kdm/*` und `/etc/X11/gdm/*`. Der xdm verwendet die Datei **Xsession**, um zu definieren, welcher Window-Manager nach dem Login verwendet werden soll. Dazu werden auch die Dateien **.xsession** und **Xresources** in den jeweiligen HOME-Verzeichnissen der User eingelesen.

Das X-Windows-System ist sehr individuell konfigurierbar. Die Applikationen, die für das X-Window-System geschrieben sind, folgen den Einstellungen für Look und Feel des Grundsystems, wie es konfiguriert wurde. Alle anderen Applikationen verwenden nur das Standard X-Toolkit, Library bietet aber die Möglichkeit, sie etwas individuell anzupassen. Die Datei **Xresources** definiert das Aussehen und Verhalten dieser Applikationen. Das Verzeichnis `/usr/X11R6/lib/X11/app-defaults` enthält viele Beispiele, die nach den Applikationen benannt sind.

Für die Applikation **XLoad** findet man folgende Datei:

```
XLoad.input:           false
LabelJustify:          left
*JumpScroll:           1
*internalBoarderWidth: 0
*showGrip:              FALSE
```

Wenn man Änderungen in einer dieser Dateien durchführt, verändert man das Aussehen und Verhalten der betreffenden Applikation für alle User. Wenn man dies allerdings nur für einen einzelnen User durchführen möchte, erstellt man die Xresource-Einträge in der Datei **Xdefaults** in dem HOME-Verzeichnis des jeweiligen Users. Die Datei `.Xdefaults` beherbergt also die individuellen Anpassungen von Applikationen für einen User.

Das X-Windows-System ist auch fähig, Programme bzw. deren Ausgabe auf andere Computer, die ein X-Windows-System laufen haben, umzuleiten. Dazu gibt es zwei Wege, einmal muss man die Umgebungsvariable **DISPLAY** anpassen. Sehen wir uns die DISPLAY-Variable an. Wir starten dazu den X-Server mit `startx` und geben in einem X-Terminal (xterm) folgenden Befehl ein:

```
#echo $DISPLAY
:0.0
```

Das bedeutet, dass alle Ausgaben des X-Servers des lokalen Rechners auf den Screen 0 und damit auf das Display 0 ausgegeben werden. Die Screenangabe wird nur in Multiscreenumgebungen benötigt. Dennoch empfehle ich die vollständige Schreibweise. Wir wollen die Ausgabe unseres Rechners auf den X-Server, der auf dem Rechner `harald.powerup.at` läuft, umleiten.

```
#export DISPLAY=harald.powerup.at:0.0
```

Wenn wir uns die DISPLAY-Variable mit dem `echo`-Befehl ausgeben lassen, erhalten wir:

```
harald.powerup.at:0.0
```

Von diesem Moment an werden alle aufgerufenen X-Clients an diesen Host bzw. X-Server weitergeleitet.

Eine weitere Möglichkeit, nicht alle Ausgaben auf diesen Rechner umzuleiten, sondern nur einzelne Applikationen, also X-Clients, besteht darin:

```
#xclient -display hostname:display.screen argumente
```

Damit der entfernte X-Server diesen Verbindungsaufbau auch akzeptiert, muss man allerdings dem X-Server mitteilen, welcher Client wie auf den X-Server zugreifen darf. Dies kann man mit dem Befehl **xhost** durchführen.

```
#xhost
```

access control enabled, only authorized clients can connect.

Wir gehen gedanklich auf den Rechner harald.powerup.at und wollen den X-Server anweisen, einen Verbindungsaufbau vom Rechner client.powerup.at zu dulden.

```
#xhost +client.powerup.at
```

client.powerup.at being added to access control list

### *Achtung*

Bitte beachten Sie, dass, wenn Sie nur `xhost +` eingeben, alle Rechner die Erlaubnis haben, sich mit Ihrem Rechner zu verbinden.

Wenn Sie nun `xhost` ohne Argumente aufrufen, erhalten Sie eine Liste mit berechtigten Clients.

```
#xhost
```

```
access control enabled, only authoriezed clients can connect.  
INET:client.powerup.at
```

Die Angabe `INET:` vor dem Rechnernamen sagt uns, dass es sich um eine TCP/IP-Verbindung handelt.

Um einen Rechner wieder aus dieser Liste zu entfernen, gibt man den selben Befehl wie beim Hinzufügen an, nur dass man das `+` durch ein `-` ersetzt.

```
#xhost -client.powerup.at
```

Das Einrichten eines Druckers unter Linux scheint auf den ersten Blick das Schwierigste zu sein. Auch auf den zweiten Blick hat sich dieser Eindruck noch immer nicht verflüchtigt. Aber beim dritten Blick – hinter die Kulissen – erkennt man, dass die Idee hinter dem kompliziert anmutenden System eigentlich sehr trivial ist. Der Administrator erhält damit alle Freiheiten bezüglich der Konfiguration, was man besonders in großen Netzwerken wirklich schätzen und lieben lernt.

Die Drucker werden unter Linux über Druckerwarteschlangen angesprochen, die von einem Druckerdaemon verwaltet werden. Jeder Drucker wird dabei von einer Warteschlange oder auch Spoolverzeichnis repräsentiert. Die Warteschlangen haben verschiedene Vorteile, so ist z. B. jeder Druckauftrag absetzbar, unabhängig davon, ob der Drucker aktiv ist oder nicht. Es geht nichts verloren. Auch ist die Trennung von zwei Druckersprachen PCL und PS eines einzigen Druckers in scharf unterschiedene Warteschlangen möglich. In diesem Kapitel werden wir uns mit der Konfiguration von Druckern, die sowohl lokal als auch über das Netzwerk ansprechbar sein sollen, befassen. Dabei lernen wir die Druckerdaemonen, die Filter sowie Druckkommandos und Administrationsprogramme kennen.

### 10.1

#### Der Herr der schwarzen Zunft – Der Druckerdaemon lpd

Die zentrale Komponente des Drucksystems unter Linux ist der **lpd** (line printer daemon). Der Druckerdaemon wird normalerweise über ein Start-Skript im entsprechenden Runlevel gestartet. Die Datei **/etc/printcap** ist die Konfigurationsdatei dazu. In ihr werden alle Drucker, die am oder besser gesagt für das System verfügbar sind, definiert. Dabei macht es keinen Unterschied, ob es sich um einen lokalen Drucker oder einen Drucker im Netzwerk handelt. Die Datei hat folgenden Aufbau.

```
Printer_name | [alias]::definitionen::last_definition;
```

Die meisten Distributionen liefern schon eigene sehr komfortable Konfigurationsprogramme für die Drucker mit. Da wir uns aber

auf Distributionsunabhängigkeit konzentrieren wollen, sehen wir wieder unter die Motorhaube.

Angaben, die in der Datei `/etc/printcap` vorkommen sollten:

**Tabelle 137: Optionen für die Datei `printcap`**

<b>Eintrag</b>	<b>Bedeutung</b>
Erster Eintrag	Name der Druckerwarteschlange.
af	Name einer Datei zum Protokollieren der Zugriffe.
lp	Lokales Ausgabegerät. So wäre hier <code>/dev/lp0</code> anzugeben, wenn der Drucker am ersten parallelen Port angeschlossen ist: LPT0.
rm	Netzwerkadresse des Druckers. Bedingt den Eintrag <code>lp=</code> .
rp	Name des Netzwerkdrukers bzw. der Name der Druckerwarteschlange auf dem entfernten Rechner (Drucker).
sd	Verzeichnis, in dem die Druckaufträge zwischengespeichert werden.
lf	Logdatei.
if	Inputfilter.
of	Outputfilter.
mx	Maximale Größe von Druckaufträgen.
sh	Unterdrückt die Statusseite.

So könnte z. B. die Datei `printcap` folgendes Aussehen haben.

```
#/etc/printcap auf seppi.powerup.at
#lokaler Drucker mit Namen lp an LPT1
lp:\
```



```
:lp=/dev/lp1:\n:sd=/var/spool/lp:\n:lf=/var/spool/lp/log:\n:af=/var/spool/lp/acct:\n:if=/var/spool/lp/mein_filter:\n:mx#0:\n:sh:\n#\n#Entfernter Drucker am Rechner 192.168.1.200 Warteschlange lp\nlp-remote:\n:lp=:\n:rm=192.168.1.200:\n:rp=lp:\n:lf=/var/spool/lp-remote/log:\n:af=/var/spool/lp-remote/acct:\n:if=:\n:mx#0:\n:sh:
```

### Einschub

Natürlich müssen die in der Datei `printcap` angegebenen Verzeichnisse existieren und die richtigen Zugriffsrechte gesetzt sein. Die meisten Distributionen haben standardmäßig einen User für den `lpd` bereits angelegt, und dieser heißt entweder ***daemon*** oder meistens ***lp***. Wir erzeugen die Verzeichnisse und setzen die Rechte für den Benutzer `lp` entsprechend.

Der Zugriff auf die Drucker kann eingeschränkt werden. Dazu editiert man die Datei `/etc/hosts.lpd`. Hier trägt man Zeile für Zeile die Namen der Workstations ein, die auf den Drucker zugreifen dürfen, inklusive der Möglichkeit, den Username dahinter noch anzugeben, der von dieser Workstation den Drucker benutzen darf. Ein Eintrag kann in etwa so aussehen:

```
perg.powerup.at helmut
```

Wichtig ist: Wenn kein Eintrag in dieser Datei vorhanden ist, haben alle darauf Zugriff.

## 10.2

### Filter über Filter aber kein kalter Kaffee

Alle Dateien, die gedruckt werden sollen, werden über so genannte Eingabe- und Ausgabefilter verarbeitet. Die Eingabefilter können eventuell das Ausgangsformat der Datei feststellen und in PostScript umwandeln. PostScript ist überhaupt DAS Format, das auf Linux-Systemen zur Anwendung kommt. Der Ausgangsfilter ist dazu da, die vom Inputfilter erzeugte Datei in die Sprache umzuwandeln, die der Drucker auch wirklich verstehen kann – z. B. PCL. Am einfachsten ist die Behandlung von PostScript-Dateien und Druckern.

Der Administrator kann seine Filter selbst schreiben. Dazu benötigt er allerdings sehr gute Kenntnisse im Shell-Skripting und beim Umgang mit den verschiedensten Dateientypen.

Einfacher ist es, bereits geschriebene, oftmals sogar vom Hersteller des Druckers oder des Programms selbst, Filter aus dem Internet zu laden. Es gibt drei populäre Druckerfilter-Pakete:

1. Apsfilter
2. Magicfilter
3. Red Hat Printer Tool

Wir wollen uns zuerst dem **apsfilter**-Paket zuwenden. Wenn es auf dem Rechner nicht bereits installiert ist, laden wir es einfach von der Web-Site [www.apsfilter.org](http://www.apsfilter.org) herunter und entpacken es. Anschließend starten wir `./configure` und `make install`, so wie wir es schon kennen.

Der Standardinstallationspfad ist `/usr/local/share/apsfilter`. In vielen Distributionen ist dieses Verzeichnis auf `/etc/apsfilter` gelegt worden. Hier finden wir die Treiberdateien und Vorlagen sowie ein **SETUP**-Programm, welches den Administrator bei folgenden Tätigkeiten unterstützt.

- Konfiguration von seriellen, parallelen und remote Druckern.
- Erstellen von notwendigen Druckerspools-Verzeichnissen.
- Generiert eine passende `printcap`-Datei.
- Setzt den `apsfilter` richtig auf.
- Druckt eine Testseite aus.
- Speichert `SETUP`-Informationen in der Konfigurationsdatei **apsfilterrc**.

Wir wollen beispielhaft einen HP LaserJet 6L in unserem System lokal definieren. Wir starten das ./SETUP-Programm und folgen den Anweisungen, die uns das textbasierte Tool (Shell-Skript) anzeigt.

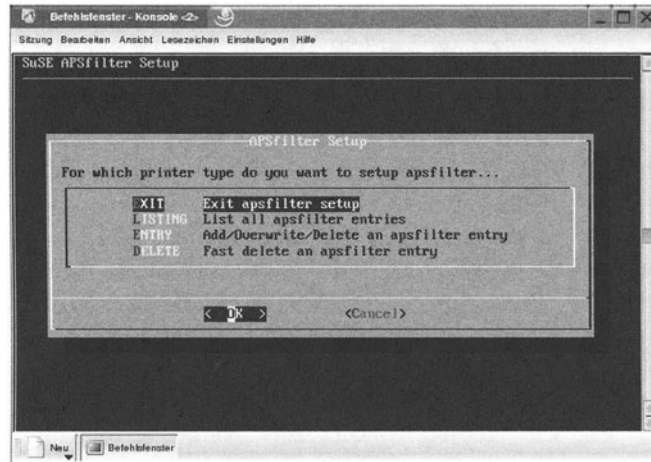


Abbildung 24: Hauptmenü vom afsfilter SETUP-Programm

Das Skript stellt fest, ob wir zuvor schon einen Drucker bzw. die Datei /etc/printcap erstellt haben und fragt uns, ob wir diese Datei überschreiben oder ob wir diesen Konfigurationsvorgang an die Datei anhängen wollen.

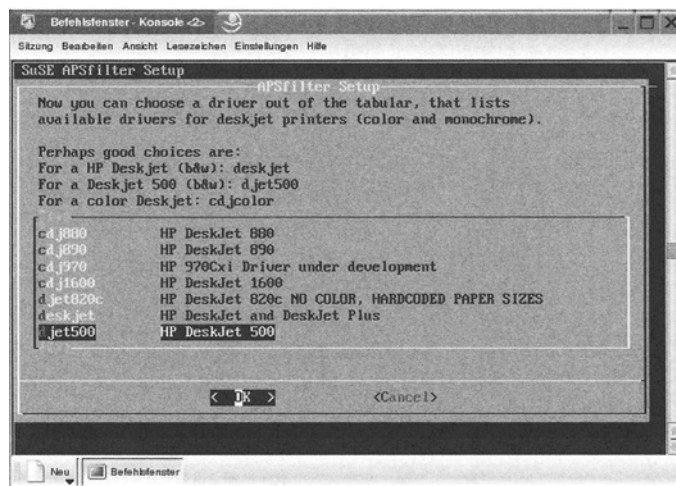


Abbildung 25: Druckerauswahl

Wir wählen zuerst einmal die Nummer 1 im Hauptmenü. Im Folgenden wählen wir die Nummer 3 – *Drucker die direkt von Ghostscript unterstützt werden*. Dort finden wir ein Menü mit Druckern vor, wo wir uns den Eintrag mit HP Laserjet 4 default, da die Treiber für den Laserjet 6 nicht funktionieren, suchen und auswählen, indem wir die Nummer eingeben. Danach werden wir nach dem Interface gefragt. Wir nehmen an, unser Drucker ist auf dem ersten parallelen Port angeschlossen, und geben deshalb den gesamten Pfad */dev/lp0* an. Wieder im Hauptmenü, wählen wir den Eintrag für die Papiergröße aus und stellen dort DIN A4 ein. Die Druckqualität belassen wir bei der Voreinstellung von medium. Nachdem unser Drucker kein Farbdrucker ist, wechseln wir den Farbmodus (Menüpunkt 5) auf grayscale (Graustufen) um. Abschließend ändern wir die Druckauflösung (6) auf 600x600 dpi.

Nun sind wir fertig und können eine Testseite drucken lassen, um zu überprüfen, ob alles so funktioniert, wie wir es wollen. Wenn die Testseite wie gewünscht und fehlerlos gedruckt wurde, haben wir auch schon gewonnen.

Unsere so erzeugte Datei */etc/printcap* hat folgendes Aussehen:

```
# /etc/printcap
#
# Please don't edit this file directly unless you know
# what you are doing!
# Be warned that the control-panel printtool requires a
# very strict format!
# Look at the printcap(5) man page for more info.
#
# This file can be edited with the printtool in the control-panel.

# APS1_BEGIN:printer1
# - don't delete start label for apsfiler printer1
# - no other printer defines between BEGIN and END LABEL
lp|ljet4;r=300x300;q=medium;c=gray;p=a4;m=auto:\
    :lp=/dev/lp0:\
    :if=/usr/local/etc/apsfilter/basedir/bin/apsfilter:\
    :sd=/var/spool/lpd/lp:\
    :lf=/var/spool/lpd/lp/log:\
    :af=/var/spool/lpd/lp/acct:\
```

```

:mx#0:\
:sh:
# APS1_END - don't delete this

```

Viele Distributionen liefern auch ein Programm namens ***print-tool*** mit, das eine grafische Konfigurationsoberfläche bietet. Sie können es aufrufen, indem Sie folgende Kommandozeile eingeben:

```
#printtool
```

Das letzte Paket ist das Magicfilter-Package. Dieses bietet keine so komfortable Konfiguration. Das bedeutet, dass wir die Datei `printcap` von Hand erstellen müssen. Das Paket kann von der Site `ftp://metalab.unc.edu/pub/Linux/system/printing` geladen werden.

## 10.3

### Ohne Manager geht gar nichts

Das Programm ***lpc*** (line printer control) ist der Herr des Druckerdaemons. Der `lpd` erhält von den Usern die Druckaufträge, und mit dem `lpc`-Programm kann man den Druckerdaemon und damit die Drucker kontrollieren. Also einzelne Druckerwarteschlangen aktivieren oder stoppen, die Reihenfolge der Abarbeitung verändern, .... . Das Druckermanagement stellt einen wichtigen Aufgabenbereich eines Systemadministrators dar. Denn oft kommt es vor, dass z. B. ein Drucker deaktiviert werden muss, weil ein Defekt des Druckers bereinigt werden muss.

**Tabelle 138: Optionen für `lpc`**

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
? <code>COMMAND</code>	Gibt einen kurzen Hilfetext für das <code>COMMAND</code> aus.
<code>abort</code> <code>all printer</code>	Terminiert einen aktiven Spooling-daemon und deaktiviert das Drucken des spezifizierten Druckers.
<code>clean</code> <code>all printer</code>	Löscht alle temporären Daten-Files und Kontroll-Files für Druckjobs, die nicht gedruckt werden können.

disale all printer	Deaktiviert die Drucker-Queues des spezifizierten Druckers.
down all printer message	Schaltet die spezifizierte Drucker-Queue aus und schreibt message in das Statusfile.
enable all printer	Aktiviert das Spoolen für den angegebenen Drucker.
exit oder quit	Verläßt das lpc-Tool, wenn im interaktiven Modus ausgeführt.
restart all printer	Versucht einen neuen Druckerdaemon zu starten.
start all printer	Aktiviert das Drucken und startet einen Spooldaemon.
status all printer	Gibt den Status des Daemons und der Queue der lokalen Maschine aus.
stop all printer	Stopt den spooling daemon nach dem aktuellen Job und deaktiviert das Drucken
topq printer jobnr user	Platziert den angegebenen Druckjob des jeweiligen Druckers an die Spitze der Queue.
up all printer	Aktiviert alles und startet einen neuen Druckerdaemon. Also das Gegenteil von down.

Wir wollen uns den Status unseres Druckersystems ansehen:

```
#lpc status
```

```
lp:
```

```
  queuing is enabled
  printing is enabled
  no entries
  printer is idle
```

Das gleiche Ergebnis hätten wir auch erhalten, wenn wir das `lpc`-Tool interaktiv ausgeführt hätten.

```
#lpc
lpc>status
lp:
    queuing is enabled
    printing is enabled
    no entries
    printer is idle
lpc>exit
```

## 10.4

### Nun bringen wir's zu Papier

Wir wissen schon einiges darüber, wie man Drucker einrichtet und verwaltet. Nun wollen wir auch etwas ausdrucken. Die meisten Programme unterstützen das Drucken direkt aus der Applikation heraus, insbesondere im X-Window-System. Das Kommando ***lpr*** dient dazu, dass man aus der Kommandozeile oder einem Skript heraus drucken kann. Das heißt, `lpr` fügt der Drucker-Queue des mit `-P` angegebenen Druckers einen Druckjob hinzu. Wird die Option `-P` nicht verwendet, wird der Druckauftrag der Queue des Druckers hinzugefügt, der in der Umgebungsvariablen ***PRINTER*** zu finden ist.

**Tabelle 139: Optionen für `lpr`**

<b>Optionen</b>	<b>Bedeutung</b>
<code>-c</code>	Dateien beinhalten Daten von <code>cifplot</code> .
<code>-d</code>	Wie oben nur von <code>tex</code> ( <code>dvi</code> -Format).
<code>-f</code>	Benutzt einen Filter, der den ersten Charakter als Fortran carriage control behandelt.
<code>-l</code>	Verwendet einen Filter, der es erlaubt, Kontroll-Charakter zu drucken und Seitenumbrüche zu unterdrücken.

-p	Verwendet das Kommando pr, um die Daten zu formatieren.
-t	Bei den Daten handelt es sich um troff-Daten.
-P DRUCKER	Diese Option leitet den Druckjob auf den Drucker DRUCKER um.
-h	Verhindert, dass die burst-Seite ausgedruckt wird.
-m	Sendet eine E-Mail, wenn der Druckauftrag abgearbeitet ist.
-s	Dabei wird in das Spoolverzeichnis nur ein Link der zu druckenden Datei gelegt. So ist es möglich, sehr große Dateien zu drucken.
-#num	Es werden nun Kopien des Druckjobs angefertigt.
-J job	Es wird der Name job auf der burst-Page ausgegeben.
-T titel	Titelname anstelle des Dateinamens für pr.
-U user	Auf der burst-page wird der Username user ausgegeben.
-wnum	Es wird num als Seitenbreite für pr verwendet.

Wir wollen das natürlich sofort testen.

```
#lpr -P lp /etc/passwd
```

Dieser Befehl sollte uns die Passwortdatei unseres Systems auf dem Drucker mit dem Namen lp ausgeben. Wenn nicht, wenn also etwas schief gegangen ist, können wir in der Datei **/var/spool/lpd/lp/log** nachsehen, ob etwas über den Fehler vermerkt wurde.

Es kann z. B. sein, dass eine Datei, die zum Drucken benötigt wird, fehlt. Wir haben schon gesehen, dass das Standardformat



für das Drucken unter Linux eigentlich PostScript ist. PostScript ist eine Entwicklung von Adobe Systems. Was verbirgt sich hinter dem Wort PostScript eigentlich? Viele sagen, dass PostScript eine Programmiersprache ist. Ich vertrete eher den Standpunkt, dass PostScript eine Seitenbeschreibungssprache ist – ähnlich HTML. Wollen wir also eine Datei, die nicht in PostScript vorliegt, ausgeben, muss sie zuerst in das PostScript Format umgewandelt werden. Diese PostScript-Datei kann direkt zu einem PostScript-fähigen Drucker geschickt werden. Oft hat man aber nicht nur PostScript-Drucker in einem Unternehmen, sondern Drucker, die unterschiedliche Druckerdialekte sprechen. Hier hilft uns aber, dass alle Druckerhersteller mit HP kompatibel sein wollen und somit fast alle Drucker auch die HP-Druckersprache PCL verstehen können. Wir benötigen deshalb ein Tool, das die PostScript-Daten in PCL-Daten umwandelt. Eine Möglichkeit dazu ist das Programm **ghostscript**. Wenn die Daten, so wie in unserem Fall, als reiner ASCII-Text vorliegt, benötigt man wiederum ein Programm, das diesen ASCII-Text in PostScript umwandelt. Dafür steht uns das Programm **a2ps** (ascii to PostScript) zur Verfügung. All diese Programme und noch weitere, werden vom `apsfilter` verwendet und eingesetzt, um korrekte Ausdrücke aus den unterschiedlichsten Programmen und Kommandos heraus zu erzeugen.

Sehen wir uns so eine PostScript-Datei an. Wir wollen dazu gleich die gerade ausgedruckte Datei `/etc/passwd` verwenden.

```
%!PS-Adobe-3.0
%%Title: passwd
%%For: root
%%Creator: a2ps version 4.13
%%CreationDate: Tue Sep 16 16:32:54 2003
%%BoundingBox: 24 24 571 818
%%DocumentData: Clean7Bit
%%Orientation: Landscape
%%Pages: 1
%%PageOrder: Ascend
%%DocumentMedia: A4 595 842 0 ( ) ( )
%%DocumentNeededResources: font Courier
%%+ font Courier-Bold
%%+ font Courier-BoldOblique
%%+ font Courier-Oblique
%%+ font Helvetica
```

```
. . .
. . .
. . .
```

### **...16 Seiten später ...**

```
. . .(Printed by root) rhead
(/etc/passwd) (1/1) (Tuesday September 16, 2003) footer
end % of isoldict
pagesave restore
showpage

%%Trailer
end
%%EOF
```

Sofort erkennen wir, dass das PostScript-Format nicht eines ist, das gerade darauf erpicht ist, kleine Dateien zu erzeugen. Allerdings ist es, und damit auch der Drucker, der Garant für einen perfekten Ausdruck in optimaler Qualität. Deshalb ist es auch das Standardformat in den Offset-Druckereien. In letzter Zeit setzt sich aber in den Druckereien das **PDF**-Format (auch von Adobe) immer stärker durch und wird bald zum Standard werden. Wir haben es alle schon vermutet – das PDF-Format setzt auf dem PostScript-Format auf.

Wenn der Ausdruck unserer `/etc/passwd`-Datei von vorhin nicht funktioniert hat, kann das daran liegen, dass z. B. das Tool `a2ps` nicht installiert ist. Diese Information würden wir aber in der Logdatei `/var/spool/lpd/lp/log` finden.

Mit dem Befehl ***pr*** können wir Dateien zum Drucken aufbereiten lassen und anschließend an das Druckerdevice oder in eine Druckdatei umleiten.

**Tabelle 140: Optionen für *pr***

<b>Optionen</b>	<b>Bedeutung</b>
-c	Druckt Steuerzeichen als hat (^D) oder als \oktal.
-d	Double Space in der Ausgabe.
-e EINZEICH TAB-WEITE	Verändert die Eingabezeichen EINZEICH in TAB-WEITE.

-h HEADER	Verwendet einen zentralen HEADER anstelle des Dateinamens.
-l LÄNGE	Setzt Leistenlänge auf LÄNGE.
-w WEITE	Setzt die Seitenbreite.
-o RAND	Fügt RAND Leerzeichen ein. Dies wird zur Seitenweite hinzuaddiert.

## 10.5

### Auch die Schlange will verwaltet werden

Nachdem wir in der Lage sind, Dateien zu drucken, werden wir nicht die Einzigen sein, die den Drucker benutzen und somit den Spoolingmechanismus bzw. die Queue verwenden.

Nachfolgend werden wir uns deshalb mit der Drucker-Queue und deren Verwaltung auseinandersetzen.

Wir gehen davon aus, dass wir zwei Druckerwarteschlangen in unserem System konfiguriert haben und zwar die Standard-Queue lp und eine ljet. Nur die ljet ist funktionstüchtig.

```
#lpc status
lp:
    queuing is enabled
    printing is disabled
    no entries
    printer idle
ljet:
    queuing is enabled
    printing is enabled
    no entries
    printer idle
#lpc stop ljet
#lpc status
lp:
    queuing is enabled
    printing is disabled
    no entries
    printer idle
ljet:
    queuing is enabled
```

```

printing is disabled
no entries
printer idle
#a2ps -Pljet /etc/passwd
#lpq status
lp:
    queuing is enabled
    printing is disabled
    no entries
    printer idle
ljet:
    queuing is enabled
    printing is disabled
    1 entry in spool area
    printer idle
#lpq
Warning: lp is down: printing disabled

```

Wir erkennen in dieser Abfolge, dass das Kommando **lpq** (line printer queue) jenes Kommando darstellt, das mit der Drucker-Queue umzugehen vermag. In unserem Fall ist es allerdings so, dass wir den Drucker gestoppt haben und so keine Anzeige mit `lpq` bekommen, jedoch finden wir mit `lpq`, dass ein Spooleintrag vorhanden ist. Schalten wir das Drucken wieder ein, dann wird der Druckjob verarbeitet.

```
#lpq start ljet
```

**Tabelle 141: Optionen für lpq**

<b>Optionen</b>	<b>Bedeutung</b>
-p	Spezifiziert einen bestimmten Drucker.
-l	Erhöht die Menge an Informationen, die ausgegeben werden. Normalerweise wird nur soviel an Informationen ausgegeben, wie in eine Zeile passt.
-a	Informationen über alle Drucker werden ausgegeben.

Wir nehmen an, dass der Drucker für die Warteschlange lp in unserem System nicht vorhanden ist. So kann natürlich jeder Druckauftrag, der dorthin geschickt wird, nicht ausgeführt werden. Damit können wir die Einträge in unserer Warteschlange besser sehen, und es motiviert uns, Jobs aus der Schlange zu löschen bzw. umzusortieren.

```
#lpr -Plp /etc/passwd
#a2ps -Plp /etc/services
#lpq
lp is ready and printing
Rank  Owner      Job  Files                Total Size
active root        8   /etc/passwd          978 bytes
1st   root        9   (standard input)     48145 bytes
#lpq -l
lp is ready and printing

root: active                                [job 008debian]
      /etc/passwd                          978 bytes

root: 1st                                  [job 009debian]
      (standard input)                     48145 bytes
```

Wir wollen, dass der Druckjob mit der Nummer 9 als erstes abgearbeitet werden soll. Wir erinnern uns, dass wir dies mit dem lpc-Kommando durchführen können.

```
#lpc topq lp 9
lp:
    moved cfA009debian
#lpq
lp is ready and printing
Rank  Owner      Job  Files                Total Size
1st   root        9   (standard input)     48145 bytes
1st   root        8   /etc/passwd          978 bytes
```

Letztendlich kommt der Benutzer, der den Job mit der Nummer 8 gesendet hat, zu uns und erklärt uns, dass er aus Versehen die falsche Datei zum Drucken geschickt hat. Er bittet uns, diesen Druckauftrag zu entfernen. Wir können ihm die Bitte natürlich

nicht abschlagen und geben auf der Kommandozeile Folgendes ein:

```
#lprm lp 8
```

```
#lpq
```

```
Rank   Owner      Job  Files              Total Size
1st    root        9    (standard input)  48145 bytes
```

**Tabelle 142: Optionen für lprm**

<b>Optionen</b>	<b>Bedeutung</b>
-PDRUCKER	Spezifiziert einen speziellen Drucker.
-	Damit werden alle Druckaufträge des Users gelöscht.
User	Diese Option ist nur für den SuperUser sinnvoll, denn er löscht alle Aufträge des angegebenen Users.
JOBNUM	Der Job mit der Nummer JOBNUM wird gelöscht.

*Bemerkung*

Am einfachsten und von der Druckqualität am besten ist es, wenn man die Mehrkosten für einen PostSkript-fähigen Drucker nicht scheut und sich ein solches Gerät anschafft. **GDI-Drucker** sind speziell für Windows-Systeme gebaut und werden standardmäßig von Linux nicht unterstützt.

---

## Shell-Skripts braucht das Land

---

Wir haben im Laufe dieses Buches schon sehr oft von Shell-Skripts direkt oder indirekt gesprochen und profitiert. Erinnern wir uns nur an die Start- und Stop-Skripts, die wir benötigen, um das System nach unseren Wünschen booten zu lassen.

Außerdem sind fundierte Kenntnisse in der Shell-Programmierung notwendig, um als Linux-Systemadministrator bestehen zu können. Wenn wir ehrlich sind, benötigt auch ein Windows-Systemadministrator einige Skripting-Kenntnisse, um verschiedene Tasks zu automatisieren. Nun habe ich auch schon die Katze aus dem Sack gelassen. Shell-Skripts sind Programme, die mit den Shell-eigenen und mit externen Programmen neue Kommandos bzw. Programme erstellen und somit auch das Linux-System um diese Funktionalität erweitern.

Mit dieser Technik kann man auch die Komplexität von vielen Programmen und Vorgängen durch Erstellung von Skripts, die interaktiv ablaufen, vereinfachen. Denken wir nur an das Kapitel 9, wo wir das X-Windows-System installiert haben. Das Kommando `./Xinstall.sh` ist nichts anderes als ein Shell-Skript. Die Endung `sh` ist nicht zwingend, aber es macht uns Humanoiden das Leben leichter, da wir sofort erkennen, dass es sich dabei um ein Shell-Skript handelt. Da die `bash`, also die Bourne Again Shell, die Standard-Shell unter Linux ist, werden wir uns hier nur mit dieser auseinandersetzen. Die grundsätzlichen Konzepte sind aber – von Syntaxunterschieden abgesehen – auf andere Shell-Varianten einfach zu übertragen.

### 11.1

#### Eine kleine Retrospektive

Wir haben schon viele Eigenschaften von Komponenten im bisherigen Verlauf kennengelernt, deshalb möchte ich dieses wichtige Kapitel mit einem Blick zurück beginnen.

Beginnen möchte ich mit den Sonderzeichen oder Metacharaktern, die in einem gewissen Kontext eine spezielle Bedeutung besitzen und die wir auch in Shell-Skripts verwenden können.

Fassen wir die bisher Gelernten und deren Bedeutung in Shell-Skripten in folgender Tabelle zusammen und lassen auch gleich das eine oder andere Neue einfließen.

**Tabelle 143: Zeichen mit besonderer Bedeutung**

<b>Zeichen</b>	<b>Bedeutung</b>
;	Kommando Trenner.
:	Dummy-Kommando - tut einfach nichts.
	Pipe, leitet stdout des einen Kommandos in stdin des nächsten Kommandos um.
&	Startet ein Programm im Hintergrund.
&&	cmd1 && cmd2 - Führt das Kommando cmd2 nur aus, wenn das Kommando cmd1 erfolgreich war.
	Wie &&, nur das cmd2 nur ausgeführt wird, wenn cmd1 nicht erfolgreich war.
.	. DATEI - Es wird der Inhalt der Datei DATEI in das aktuelle Script eingefügt.
#	Kommentarzeichen.
#!/bin/sh	Das Script wird mit dieser Shell ausgeführt. Da ein # zu Beginn der Zeile steht, ist es eigentlich ein Kommentar, aber der Kernel erkennt daraus das Dateiformat, außerdem ist es für uns ein Hinweis, welche Shell wir zur korrekten Ausführung benötigen werden.
?	Ein beliebiges Zeichen.
*	Beliebig viele beliebige Zeichen.
<	Eingabeumleitung überschreibend.
>	Ausgabeumleitung überschreibend.
>>	Ausgabeumleitung anhängend.
<< ENDE	Liest aus einer Datei bis ENDE.
>&	Umleitung von stdout und stderr.



[abc]	Entweder a oder b oder c.
( )	Kommandos innerhalb der Klammer werden in einer Shell ausgeführt.
{ }	Damit werden Kommandos gruppiert.
{ , , }	Zeichenketten zusammensetzen.
\	Hebt die Bedeutung des nachfolgenden Zeichens auf. Bsp. \? definiert ein ? und nicht ein beliebiges Zeichen.
\$	Variablen-Inhaltsoperator.
\$* oder @\$	Liste der Parameter, die einem Kommando übergeben wurden.
\$#	Anzahl der Parameter, die einem Programm übergeben wurden.
\$0	Name des Programmes.
\$1 bis \$9	Die Parameter 1 bis 9.
\$( )	Kommandosubstitution.
\${ }	Funktion zur Manipulation von Zeichenketten.
\$( )	Berechnung von arithmetischen Ausdrücken.
“ ”	Auswertung von gewissen Sonderzeichen verhindern.
, , ,	Auswertung sämtlicher Sonderzeichen werden verhindert.
` `	Kommandosubstitution.
[ AUSTR ]	Eine andere Schreibweise für den test-Begriff.
~	Steht für das HOME-Verzeichnis eines Users. (Bekannt von manchen URL's).

Wir wissen schon, dass ein Shell-Skript, will man es ausführen, das Execute-Recht besitzen muss. Damit als User helmut der Kommandoaufruf ./script-name funktioniert, muss das Shell-

Skript `script-name` für den User das Zugriffsrecht „ausführbar“ bereitstellen. Dies kann einerseits über die Eigentümerschaft, die Gruppenmitgliedschaft oder über den Bereich „Rest der Welt“ definiert werden. In einem der drei angegebenen Bereiche muss das `x` für Execute in den File-Permissions erscheinen. Alternativ kann man das Shell-Skript auch mittels `#bash script-name` aufrufen. Hierbei muss das Skript für den User nur lesbar sein. Es benötigt also kein Execute-Recht.

### *Hinweis*

Beachten Sie die besondere Bedeutung von SUID und SGID.

Wir beschränken uns bei unseren Betrachtungen auf die `bash`. Bei den meisten Distributionen ist die `sh` ein Link auf die `bash` so, dass wir im Weiteren die beiden Shells `sh` und `bash` synonym verwenden wollen. Unsere Kenntnisse über den `vi` kommen uns jetzt auch sehr zu Gute.

## 11.2

### Hello World

Jedes Buch oder jeder Beitrag über das Programmieren fängt mit dem allseits berühmten „Hallo Welt“-Programm an. Wir beginnen hier auch damit und geben auf der Kommandozeile Folgendes ein:

```
#vi hallo.sh
```

In die Datei geben wir nun unser erstes Programm – unser erstes Shell-Skript ein.

```
#!/bin/sh
echo "Hallo Welt!"
```

Mit `:wq` speichern wir die Datei und beenden den Editor. Nun setzen wir die Ausführbarkeitsrechte so, dass nur wir – `root` – das Recht haben, dieses Programm auszuführen.

```
#chmod 700 hallo.sh
```

Lassen Sie uns das Script starten:

```
#./hallo.sh
Hallo Welt!
#
```

Wir erinnern uns wieder an die `PATH`-Umgebungsvariable. Dort ist enthalten, in welchen Pfaden unser System nachsieht, ob der eingegebene Ausdruck ein gültiger Befehl ist oder nicht. Wir

wollen uns ein eigenes Verzeichnis für unsere selbstgeschriebenen Skripts erstellen und dieses dauerhaft in unsere Umgebung integrieren.

#### *Hinweis*

Sie sollten schon selbstständig in der Lage sein, dies durchzuführen. Probieren Sie es, bevor Sie weiterlesen.

Wir erinnern uns an die Konfigurationsdateien der Shell – wo die Umgebungsvariablen, also das Environment gesetzt werden.

Wir sind schon auf die Dateien `.bashrc` und `.profile` sowie `.xdefaults` und `.xsession` eingegangen. Dateien, die wir noch finden können, sind **`.bash_login`** und **`.bash_logout`**. Jedesmal, wenn wir uns an das System anmelden, wird die Datei `.bash_login`, falls vorhanden, abgearbeitet. Es ist unschwer zu erraten, welche Datei jedes Mal ausgeführt wird, wenn wir uns vom System abmelden.

Wir haben damit noch einmal den Unterschied herausgearbeitet von `su` und `su -`. Bei der zweiten Art werden auch die Login-Skripts abgearbeitet, bei der ersten Art nicht.

Wir wollen den Pfad für unser neues Verzeichnis **`/scripts`** in unseren Pfad dauerhaft aufnehmen. Wir haben die Qual der Wahl, ob wir dies in `.profile` in `.bashrc` oder in `.bash_login` bewerkstelligen wollen. Nachdem `.bash_login` bisher ein benachteiligtes Leben geführt hat, wollen wir sie jetzt benutzen. Wir rufen die Datei `~/.bash_login` auf und fügen die Zeile `export PATH=$PATH:/scripts` hinzu und speichern sie wieder ab. Nun wird dieser Pfad jedes Mal, wenn wir uns am System anmelden, gesetzt. Für die anderen Dateien und das Setzen und Löschen von Aliases verweise ich Sie auf Kapitel 2 und 3.

Wie Sie vermuten, handelt es sich bei diesen eben angesprochenen Dateien ebenfalls um Shell-Skripts, und wir haben damit schon wieder eine wichtige Tätigkeit des Systemadministrators durchgeführt – nämlich bestehende Skripts anpassen.

## 11.3

### Der Werkzeugkoffer

Wir wollen nun unseren Werkzeugkoffer bestücken, damit wir „more sophisticated“ Shell-Skripts bauen können. Shell-Skripting kann man am besten anhand von Beispielen lernen. Deshalb werden wir einige Beispiele und die darin verwendeten Konstrukte beleuchten. Ich lege Ihnen auch nahe, dass Sie so viele Shell-Skripts wie möglich von anderen analysieren und zu verste-

hen versuchen, wie die Programmierer welche Aufgaben lösen. Man kann so sehr viele Tricks und Tipps erhalten, um „schönere“ Programme zu schreiben. Dennoch sollte man nicht vergessen, dass Shell-Skripts eigentlich in die Kategorie „Quick and Dirty“ fallen.

Betrachten wir folgendes Beispiel:

```
#cat wisch_und_weg.sh
#!/bin/sh
cat /dev/null > /var/log/messages
echo "Alles Sauber!"
```

Wir können dieses Skript, da wir unseren Pfad beim Login anpassen lassen, von überall im System ausführen lassen. Dennoch sollte man sich immer über die Tragweite seines Tuns im Klaren sein. Wenn wir dieses Skript nun aufrufen, dann passiert was? Das Skript wird von oben nach unten zeilenweise ausgeführt. Es verhält sich genau so, als ob wir Befehl für Befehl eingeben würden. Das heißt aber umgekehrt, wenn Sie Tag für Tag immer die gleiche Reihenfolge von Befehlen abarbeiten, dann fassen Sie diese doch einfach zu einer Datei zusammen, und Sie brauchen nur noch dieses Skript zu starten.

Zuerst leiten wir den Inhalt des Gerätes Null – unser Datennirvana – auf die Datei /var/log/messages – unsere Hauptlogdatei – um. Da die Daten aus dem Nirvana kommen, erhalten wir eine leere Datei. Wir können die Datei messages und die meisten Logdateien nicht mit rm und touch auf Null zurücksetzen, da ein offener **Dateihandel** besteht. Wenn aber diese Logdateien sehr groß – zu groß – werden, müssen wir sie zurücksetzen. Wir haben dazu den Befehl logrotate kennengelernt. Im Prinzip leistet unser kleines Skript Ähnliches, nur gehen die Daten der Datei dabei verloren.

Wir wollen unsere beiden Skripts etwas erweitern. Der Text, der auszugegeben ist, und die Datei, die zurückgesetzt werden soll, werden als Parameter oder Argument an das Skript übergeben.

Unser Skript soll folgendermaßen aufgerufen werden können.

```
#hallo.sh "Neuer Test"
Neuer Test
#
```

*Frage*

Warum muss man Neuer Test unter “ “ eingeben?

Das Leerzeichen fungiert, wie wir wissen, als Trenner bei einer gültigen Kommandozeile.

Die Anpassung, die wir an dem Skript hallo.sh vornehmen müssen, sieht folgendermaßen aus:

```
#!/bin/sh
echo $1
```

Die Ausdrücke, die mit **\$** beginnen, werden als Variable behandelt – also als Platzhalter für die eigentlichen Werte. Die vordefinierten Variablen **\$0** bis **\$1** haben besondere Bedeutungen. Die Variable **\$1** ist das erste Argument, das an den Programmnamen (dieser wäre **\$0**) angehängt wurde. In unserem Fall der Inhalt „Neuer Test“.

Die Änderungen in unserem zweiten Skript sind dann:

```
#cat wisch_und_weg.sh
#!/bin/sh
cat /dev/null > $1
echo "Alles Sauber!"
```

Nun würde jede Datei, die an den Programmaufruf angehängt wird, gelöscht und, wenn sie nicht existiert, eine leere Datei erstellt. Beachten Sie bitte diese Feinheit. Wir testen weder, ob diese angegebene Datei existiert, ob sie schreibbar oder ein Verzeichnis noch irgendetwas anderes ist. Dies motiviert uns, Ausschau zu halten, ob es solche Testmöglichkeiten gibt und wie man diese anwenden kann. Des Weiteren wollen wir nicht alle Dateien zulassen, sondern nur gewisse Logdateien, die wir in unserem Skript hinterlegen werden.

Wir benötigen also den Begriff Ablaufsteuerung. Wir erstellen dabei Befehlszeilen, die ein Entscheidungskriterium prüfen und einen Rückgabewert produzieren, den wir als Entscheidungsgrundlage verwenden können. Wir erinnern uns wieder an folgende Eigenschaft der Shell: „Ist die Ausführung des Programmes korrekt verlaufen, bekommen wir den Rückgabewert 0, ansonsten einen der ungleich Null ist – meistens 1“. Ein einfaches Beispiel dafür ist:

```
#cmd1 && cmd2
```

Hier wird, wie wir wissen, **cmd2** nur dann ausgeführt, wenn **cmd1** einen Rückgabewert von Null hat, also ohne Fehler beendet wurde.

Dieses Ergebnis kann nun abgefragt werden und zwar mit der **if** **else** Anweisung.

```
if cmd_1
then cmd_2
else cmd_3
fi
```

Wobei zu beachten ist, dass der `else`-Teil nur optional ist. Dieser Teil muss nicht notwendigerweise angeführt werden. Manches Mal hat man aber die Aufgabe, noch komplexere Abfragen zu realisieren. Dazu kann man den `else`-Teil zusätzlich erweitern, und zwar um ein **elif then**-Konstrukt.

```
if cmd_1
then cmd_2
elif cmd_3
then cmd_4
else cmd_5
fi
```

Natürlich kann man diese Konstruktion beliebig tief verschachteln. Allerdings muss man sich des logischen Hintergrunds und der Bedeutung des Geschriebenen im Klaren sein:

```
if [[ "$1" = "$2" ]]
then
echo " $1 und $2 sind identisch"
else
echo "Die Eingaben sind ungleich"
fi
```

Mit diesem Beispiel betrachten wir, ob die angegebenen Parameter beim Programmaufruf übereinstimmen.

### **Achtung**

Die angegebenen Leerzeichen bei der `if`-Zeile sind wichtig und notwendig! Ansonsten erhalten Sie Fehlermeldungen.

Sie fragen sich jetzt sicherlich, woher die zwei eckigen Klammern in der if-Anweisung kommen. Wenn wir die obige Aussagen richtig verstanden haben, dann sollte da eigentlich nur eine stehen. Sie haben Recht. Allerdings müssten wir dann das = mit \ entwerten. Ohne den \ würden wir Fehlermeldungen bekommen. Man kann sich mit einer doppelten Klammer behelfen, und wir müssen = nicht mehr entwerten.

```
if [ "$1" \= "$2" ]
then
    echo " $1 und $2 sind identisch"
else
    echo "Die Eingaben sind ungleich"
fi
```

Auch diese Schreibweise ist korrekt, aber in meinen Augen etwas umständlicher.

Wir ändern das Skript so ab, dass wir ein „geheimes“ Wort erraten müssen. Wir hinterlegen also eine „gültige Eingabe“.

```
richtig="Hallo"
if [[ "$1" = "$richtig" ]]
then
    echo "Der Angegebene Parameter ist korrekt"
else
    echo "Die Eingabe ist nicht erlaubt"
fi
```

Nur wenn wir jetzt das Programm mit dem „richtigen“ Parameter aufrufen, wird die Abfrage korrekt behandelt. Wenn wir mehrere richtige Angaben prüfen wollen, wiederholen wir einfach das if-Konstrukt:

```
richtig1="Hallo"
richtig2="Hallo"
if [[ "$1" = "$richtig1" ]]
then
    echo "Der Angegebene Parameter ist korrekt"
elif [[ "$1" = "$richtig2" ]]
then
```

```
        echo "Der Angegebene Parameter ist korrekt"
    else
        echo "Die Eingabe ist nicht erlaubt"
    fi
```

Damit können wir unsere Aufgabenstellung beim Zurücksetzen der Logdateien erfüllen. Wir schreiben einfach unser Skript von oben um:

```
#cat wisch_und_weg.sh
#!/bin/sh
log1="/var/log/messages"
log2="/var/log/lpr.log"
if [[ "$1" = "$log1" ]]
then
    cat /dev/null > $log1
    echo "Alles Sauber!"
elif [[ "$1" = "$log2" ]]
then
    cat /dev/null > $log2
    echo "Alles Sauber!"
fi
```

Wir können auf diese Weise nur noch die von uns gewünschten Dateien zurücksetzen. Es ist eine gute Sache, nicht nur für uns, sondern für jeden, der mit diesem Programm umgehen soll, dass wir eine minimale Hilfe integrieren. Diese Hilfe soll uns die Verwendung des kleinen Skripts verdeutlichen.

Wir fügen also folgende Zeilen hinzu:

```
#cat wisch_und_weg.sh
#!/bin/sh
log1="/var/log/messages"
log2="/var/log/lpr.log"
if [[ "$1" = "$log1" ]]
then
```



```

cat /dev/null > $log1
echo "Alles Sauber!"
elif [[ "$1" = "$log2" ]]
then
cat /dev/null > $log2
echo "Alles Sauber!"
else
echo "Die erlaubten Dateien sind:"
echo "/var/log/messages"
echo "/var/log/lpr.log"
fi

```

Damit wird uns immer bei einer Falscheingabe mitgeteilt, welche Dateien man mit diesem Tool zurücksetzen kann.

Wir haben im `if`-Statement eigentlich schon einen Test verwendet. Tests haben das Schlüsselwort **test** oder die Kurzform `[ ]`. Mit einem Test kann man den Wahrheitswert einer Abhängigkeit über deren Rückgabewert feststellen.

Der Befehl `test` unterstützt neben Datei und Dateisystemoperationen auch numerische- und Zeichenkettenvariablen.

**Tabelle 144: Mögliche Tests**

<b>Test</b>	<b>Bedeutung</b>
<code>[ z1 = z2 ]</code>	Zeichenkettenvergleich auf Gleichheit.
<code>[ z1 != z2 ]</code>	Zeichenkettenvergleich auf Ungleichheit.
<code>[ -z zk ]</code>	Testet auf leere Zeichenkette.
<code>[ -n zk ]</code>	Testet auf Zeichenketten ungleich Null.
<code>[ 0 operator P ]</code>	Numerischer Test. Als Operator sind zulässig: <code>-eq</code> Gleichheit <code>-ne</code> Ungleichheit <code>-gt</code> Größer <code>-lt</code> Kleiner <code>-ge</code> Größer Gleich <code>-le</code> Kleiner Gleich

[ operator Datei ]	<p>Dateitests. Als Operator sind zulässig:</p> <ul style="list-style-type: none"> <li>• -s Datei existiert und ist nicht leer</li> <li>• -f Normale Datei und kein Verz.</li> <li>• -d Ein Verzeichnis</li> <li>• -w Schreibrecht gesetzt</li> <li>• -r Leserecht gesetzt</li> <li>• -x Ausführbarkeitsrecht ist gesetzt</li> </ul>
--------------------	---

Diese Operationen und Tests können wir nun auch mit **!** (Negation), **-a** (AND) und **-o** (ODER) zu komplexeren Ausdrücken zusammensetzen.

[ ausdruck -a ausdruck ] ----- logisches UND

Mit diesen Tests können wir z. B. Dateien anlegen und wieder löschen. Das folgende Programmfragment hat zwar keinen Sinn, zeigt aber die Verwendung von Tests sehr gut. Wir wollen im Verzeichnis **/tmp** prüfen, ob eine Datei vorhanden ist. Wenn sie existiert, soll sie gelöscht werden, und wenn sie nicht existiert, soll sie erstellt werden.

```
if [ -f "/tmp/test" ]
then
    rm /tmp/test
else
    touch /tmp/test
fi
```

Nun könnten wir unser `wisch_und_weg.sh`-Programm mit der einen oder anderen zusätzlichen Abfrage noch komfortabler und vor allem sicherer machen.

Oft soll ein Programm ein und dieselbe Aufgabe mehrere Male hintereinander ausführen. Dazu verwendet man sogenannte **Schleifen**. Bei den Schleifen gibt es mehrere Varianten, die sich von der Art der Ausführung und der Überprüfung der Abbruchbedingung unterscheiden. Die Abbruchbedingung ist besonders

wichtig, und man muss genau darauf achten, dass eine solche auch erreicht werden kann. Ansonsten hat man eine sogenannte Endlosschleife. Das heißt, unser Programm kommt aus der Schleife nicht mehr heraus und ist somit nicht fähig, das Programm weiter fortzusetzen.

Beginnen wir mit der Zählschleife **for do**. Die einfachste Form der Zählschleife ist:

```
for i
do
    echo $i
done
```

Das **i** steht hier für den **Schleifenzähler**. Solange im Schleifenzähler Werte vorhanden sind, wird die Schleife wiederholt und mit dem nächsten Wert für **i** durchlaufen. In unserem Fall kommt hier die Liste der Werte für **i** aus dem Programmaufruf. Wenn wir das Skript `for.sh` folgendermaßen aufrufen:

```
#for.sh 2 5 1
2
5
1
#
```

Als Werte für den Schleifenzähler dienen die angegebenen Parameter. Wenn kein Parameter mehr vorhanden ist, ist auch das Schleifenabbruchkriterium erreicht.

Wenn wir eine Liste der Dateien im aktuellen Verzeichnis ausgeben wollen, können wir dies auch mit der Zählschleife realisieren und zwar folgendermaßen:

```
for i in *
do
    if [ -f $i ]
    then echo $i
    fi
done
```

Mit dem Schlüsselwort `in` und dem Metazeichen `*` erhalten wir eine Liste der Namen in dem aktuellen Verzeichnis, die wir aber nur ausgeben, wenn es sich um eine Datei handelt.

Mit dem Schlüsselwort `in` können wir auch manuell eine Liste angeben, also folgendermaßen:

```
for i in 2 3 5
do
    echo $i
done
```

Als Ausgabe erhalten wir dann wie erwartet:

```
#for2.sh
2
3
5
#
```

Die nächste Art von Schleife ist die ***while do*** Schleife. Die Syntax der `while`-Schleife:

```
while cmd_1
do
    cmd_2
done
```

Diese Schleife heißt auch kopfgesteuerte Schleife, da die Überprüfung der Abbruchbedingung am Eingang der Schleife erfolgt. So wird die Schleife auch nur dann betreten, wenn die Abbruchbedingung nicht erfüllt ist. Solange also `cmd_1` erfüllt ist, wird `cmd_2` ausgeführt.

Betrachten wir folgendes Beispiel `stat.sh`, das eine Statistik über die angemeldeten User erstellt und nicht von selbst abbricht.

```
#!/bin/sh
#
# Aufruf: stat <DATEI>
#
```

```
if [ $# = 0 ]
then echo "Aufruf: stat <dateiname>"
else
    while sleep 240
    do
        date >> $1
        who >> $1
    done
fi
```

Wir rufen das Programm auf, und der Prompt kehrt nicht wieder zurück. Wir müssen das Programm also mit **STRG-C** beenden. Es liegt nahe, solche Programme im Hintergrund zu starten. Wir müssten Sie aber wieder mit dem `kill`-Kommando beenden.

Zu Beginn des Skripts überprüfen wir mit `$#` die Anzahl der angegebenen Argumente. Ist diese Null, dann wird eine Benutzungshilfe ausgegeben.

Beim Schleifeneingang wird der Befehl `sleep 240` angegeben, der immer nach 240 sec erfüllt ist und damit die Schleife ewig wiederholt, da nie eine Abbruchbedingung erlangt wird.

Die Datei, die wir beim Aufruf des Skripts angeben, muss schon bestehen. Wir könnten dieses Programm aber so anpassen, dass es zuerst die Existenz der Datei überprüft, sie auf ein Archiv kopiert und dann zurücksetzt oder, wenn sie nicht vorhanden ist, anlegt. Dies sollten Sie als Übung versuchen.

Die Ausgabe unseres Skripts, das eigentlich nur alle 240 sec ausgeführt wird, ist:

```
#cat stat.dat
Thu Sep 18 11:43:25 CEST 2003
root    pts/0      Sep 18 08:29 (:0)
root    pts/1      Sep 18 08:29 (:0)
root    pts/2      Sep 18 09:12 (:0)
root    pts/3      Sep 18 11:38 (:0)
Thu Sep 18 11:47:25 CEST 2003
root    pts/0      Sep 18 08:29 (:0)
root    pts/1      Sep 18 08:29 (:0)
root    pts/2      Sep 18 09:12 (:0)
root    pts/3      Sep 18 11:38 (:0)
```

```
Thu Sep 18 11:51:25 CEST 2003
root    pts/0          Sep 18 08:29 (:0)
root    pts/1          Sep 18 08:29 (:0)
root    pts/2          Sep 18 09:12 (:0)
root    pts/3          Sep 18 11:38 (:0)
```

Die letzte Art von Schleife ist die ***until do***-Schleife.

```
until cmd_1
do
    cmd_2
done
```

Solange `cmd_1` nicht erfüllt ist, wird die Schleife durchgeführt. Erst wenn das `cmd_1` ein erfolgreiches Ergebnis bringt, wird die Schleife abgebrochen. Wir erkennen den Unterschied zur `while`-Schleife. Der Logikansatz ist bei diesen Schleifen entgegengesetzt. Bei der `while`-Schleife wird die Wiederholung so lange fortgesetzt, bis die Bedingung falsch wird, und bei der `until`-Schleife wird die Wiederholung solange fortgesetzt, wie die Bedingung falsch ist.

Wenn wir also ein kleines Monitoring-Programm schreiben wollen, das überwacht, ob und wann sich ein gewisser User angemeldet hat, dann geben wir Folgendes ein und speichern es unter `monitor.sh` ab:

```
if [ $# != 1 ]
then
    echo "until <Benutzer_Name>"
    exit 1
else
    until who | grep $1
    do
        sleep 60
    done
fi
```

Wir haben hier einen ähnlichen Aufbau wie bei unserem vorigen Beispiel gewählt. Uns liegt eine Schleife vor, die solange

„schläft“, bis sich der User mit dem übergebenen Parameternamen am System anmeldet.

Eine weitere wichtige Art der Ablaufsteuerung ist die **case**-Anweisung. Mit diesem Konstrukt kann man Fallunterscheidungen verwirklichen. Die allgemeine Syntax lautet:

```
case var in
    muster) comd_1 ;;
    muster) comd_1 ;;
    muster) comd_1 ;;
    ...
    muster) comd_1 ;;
esac
```

Wir wollen im Kommenden feststellen, in welcher Weise ein Linux-Rechner mit Wochentagen umgeht. Wenn wir an unser Skript eine Zahl von 0 bis 6 übergeben, soll es uns sagen, ob Montag, Dienstag, u.s.w. ist.

Wir geben Folgendes ein und speichern es unter `woche.sh` ab:

```
case $1 in
    1) echo Montag
        ;;
    2) echo Dienstag
        ;;
    3) echo Mittwoch
        ;;
    4) echo Donnerstag
        ;;
    5) echo Freitag
        ;;
    6) echo Samstag
        ;;
    0) echo Sonntag
        ;;
esac
```

Lassen Sie uns das Skript testen:

```
#woche.sh 7
#
#woche.sh 4
Donnerstag
#
```

Wie wir uns denken können, ist es in allen bisher besprochenen Beispielen und Konstrukten auch möglich, RegEx zu verwenden und mit logischen UND-, ODER- und NICHT-Operatoren zu verknüpfen.

Dies ist aber schon ein Bereich, den man unter „Advanced Shell-Skripting“ zusammenfassen könnte und den Rahmen hier sprengen würde.

Abschließend wollen wir noch Shell-Funktionen behandeln. Funktionen sind, etwas banal ausgedrückt, Unterprogramme, eine Black Box von Programmcodes, die bei Aufruf etwas Bestimmtes ausführen. Ein Funktionsaufruf unterbricht die Ausführung eines Programms an der aktuellen Stelle, springt in die Funktion, führt diese aus und kehrt an die aufrufende Stelle im eigentlichen Programm zurück. Einer Funktion kann man auch Parameter übergeben, mit der sie arbeiten kann. Eine Funktion kann auch einen Rückgabewert liefern. Beides muss aber nicht sein. Die allgemeine Syntax sieht wie folgt aus:

```
funktions_name ( )
{
  Kommandos
  . . .
}
```

Die Funktionen werden im selben Skript definiert und sind auch nur in dieser einen Shell existent. Aufgerufen wird die Shell-Funktion einfach durch Angabe des Funktionsnamens. Sehen wir uns ein kleines Beispiel an, das nicht viel tut außer einen Text auszugeben.



```
#Die Funktionsdefinition muss vor dem Aufruf stehen
func ()
{
    echo "Ich bin Text aus der Funktion"
}

#Nun erfolgt der Funktionsaufruf

func
echo ""
echo "Und ich bin Text ausserhalb der Funktion"
```

## 11.4

### Ein kleiner Shell-Nachschlag

Wir haben zwar der Shell schon ein eigenes Kapitel gewidmet, aber die Shell ist so mächtig und vielfältig, dass man nicht umhin kommt, den einen oder anderen Teil noch einmal zu wiederholen oder einen Teilbereich zu präzisieren. Wir wollen uns hier noch einmal mit den Umgebungsvariablen der Shell, dem Environment, beschäftigen, also mit Shell-Variablen.

Wir haben gesehen, dass die Shell unter verschiedenen Variablen unterscheidet. Variablen können auch eine Serie von Variablen jeden Typs enthalten. Man spricht dann von sogenannten Arrays.

Eine wichtige Konfigurationsdatei für das Verhalten der Kommandozeile bzw. der Shell ist die Datei ***inputrc***. Die Datei ***inputrc*** ist wieder in zwei Varianten zu finden. Einmal als ***.inputrc*** im jeweiligen HOME-Verzeichnis des Users und einmal als globale Datei in ***/etc/inputrc***. Ist die ***.inputrc*** nicht vorhanden, wird die globale verwendet. Die Datei ***inputrc*** konfiguriert die sogenannte ***GNU readline Library***, also das Verhalten der Shell bzw. der Programme, die ein Kommandozeilen-GUI zur Verfügung stellen.

Beispiel einer ***/etc/inputrc***-Konfigurationsdatei:

```
#cat /etc/inputrc
# Be 8 bit clean
set input-meta on
set output-meta on
#
```

Die Optionen, die für eine Shell gelten, kann man mit dem sogenannten built-in-Kommando **set** definieren. Mit `set +o option` wird eine Option eingeschaltet und mit `set -o option` wieder deaktiviert.

**Tabelle 145: Optionen für set**

<b>Optionen</b>	<b>Bedeutung</b>
Ohne	Alle Shell-Variablen werden ausgegeben.
allexport	Exportiert alle erzeugten Variablen.
emacs	Die Kommandozeile kann mit emacs-Befehlen editiert werden.
vi	Die Kommandozeile kann mit vi-Befehlen editiert werden.
errexit	Wenn ein einfaches Kommando mit einem Nicht-Null-Wert terminiert wird.
hashall	Erinnert sich an den Ort der Programme und beschleunigt so die nochmalige Ausführung.
histexpand	Aktiviert die ! History-Expansion. Default is "on".
history	Aktiviert den History-Mechanismus
noclobber	Verhindert Dateiumleitungen <,>,...
noexec	Liest zwar Befehle ein, führt sie aber nicht aus.
noglob	Wildcardunterstützung wie * und ? wird deaktiviert.
ignoreeof	Ignoriert das Dateiendezeichen STRG-D.

Mit dem Wissen dieses Kapitels heiß gemacht, können wir es kaum erwarten, ein Shell-Skript zu schreiben, das auch etwas „Vernünftiges“ für uns Administratoren leistet. Wie wäre es mit einem einfachen Backupskript, das zu bestimmten Zeiten ablaufen soll.

Die Hauptaufgabe eines Systemadministrators, egal ob Windows, Linux oder irgendein anderes beliebiges Betriebssystem, liegt eindeutig in Routinearbeiten. Also Tätigkeiten, die jeden Tag immer wieder auf die gleiche oder ähnliche Art ablaufen sollen. Die Kontrolle von Log-Dateien wäre so eine Routinetätigkeit oder das Anlegen eines Backups. Wir haben Aufgaben zu erledigen, die periodisch oder zu gewissen Zeiten ausgeführt werden sollen. Zu diesem Zweck bietet uns Linux eine Unzahl von Tools und Konzepten an, damit wir uns als Administratoren das Leben leichter machen und die Routinearbeiten dem Computer überlassen können und wir für neue Projekte und Ziele einsatzbereit sind.

Die klassischen Methoden und Tools für die automatische Ausführung von Aufgaben werden wir nun im Folgenden kennen lernen.

### 12.1

#### Auf den richtigen Zeitpunkt kommt es an

Wir müssen bei allem Automatisierungswillen unterscheiden, ob die jeweilige Aufgabe nur einmal zu einer bestimmten Zeit ausgeführt werden soll oder ob sie periodisch zu gewissen Zeiten immer wieder ausgeführt werden muss. Für jede dieser Varianten stellt uns Linux das probate Mittel zur Verfügung.

Betrachten wir den Fall, dass ein sehr zeitaufwendiger Prozess, der auch sehr viele Ressourcen verbraucht, auszuführen ist. Wir können uns dabei das Reorganisieren einer nicht relationalen Datenbank in einer großen Bibliothek vorstellen. Nennen wir den Job für das Reorganisieren doch einfach **reorg.sb**. Anhand der Endung erkennen wir sofort, dass es sich dabei um ein Shell-Skript handelt.

Nun soll der Job aufgrund seiner Komplexität und seines Ressourcenhungers nicht während der Produktionsstunden laufen, sondern um 23:30 Uhr, am Samstag, den 27.9.2003, abgearbeitet werden.

Wir haben nun die Wahl, ob wir Samstag Nacht im Büro oder im privaten Rahmen verbringen wollen. Mir persönlich fällt die Entscheidung nicht schwer, deshalb wende ich mich sofort dem Tool **at** zu. Mit **at** kann man Jobs zu bestimmten Zeiten starten lassen. Die Optionen, die man dem **at**-Kommando mitgeben kann sind:

**Tabelle 146: Optionen für at**

<b>Optionen</b>	<b>Bedeutung</b>
-q	Benutzt eine definierte Queue. Diese Queues werden von a bis z oder A bis Z bezeichnet. Die Queue a ist die Defaultqueue.
-m	Wenn der Job abgearbeitet ist, bekommt der User eine Mail zugesandt.
-f DATEI	Liest den Job von der Datei DATEI anstelle der Standard-Eingabe.
-l	Listet die Jobs aller Benutzer.
-d	Löscht einen at-Job.
-v	Zeigt die Ausführungszeit des Jobs an.

Mit dem **at**-Kommando kann man sehr komplexe Zeitangaben realisieren. So wird die Zeitangabe in der Form **HH:MM** für eine gewisse Uhrzeit des jeweiligen Tages akzeptiert oder auch Schlüsselwörter wie **midnight**, **noon** oder **teatime** (16:00 Uhr). **AM** und **PM** werden genauso akzeptiert wie Zeitadditionen. Die Angabe von ganzen Daten ist ebenso erlaubt.

Am besten ist, wir sehen uns ein paar Beispiele an:

```
#at 4pm +3 days
```

Startete den Job um 16:00 Uhr in 3 Tagen.

```
#at 10am Jul 31
```

Startet den Job am 31 Juli um 10 Uhr vormittags.

```
#at 1am tomorrow
```

Dieser Job wird dann um 1 Uhr in der Früh am nächsten Tag ausgeführt.

*Hinweis*

Wenn die eingegebene Uhrzeit (wenn diese alleine steht) bereits verstrichen ist, wird der Job am nächsten Tag zur angegebenen Uhrzeit ausgeführt.

Wie sieht nun die Zeitangabe für unser Vorhaben der Reorganisation aus?

```
#at 11:30pm 27.09.03
```

```
at> reorg.sh
```

```
at> STRG+D
```

```
at> <EOT>
```

Wir sehen, dass wir mit dem `at`-Kommando in einen interaktiven Dialog einsteigen, in dem wir unsere Befehle eingeben können. Die Eingabe beenden wir mit dem End Of Text (**STRG+D**)-Zeichen. Am Rande sei hier noch bemerkt, dass die Abarbeitung des Jobs unabhängig von der Shell ist, an der wir das `at`-Kommando eingegeben haben.

```
#at -l
```

```
4 2003-09-27 a root
```

Damit erhalten wir eine Liste aller Jobs, die mit `at` abgesetzt wurden. Die Angaben sind Jobnummer, Datum, Queue und Benutzer.

Wenn wir in die Manpages des `at`-Befehls sehen, merken wir, dass es sich dabei eigentlich um eine ganze Befehlsfamilie handelt. Das gleiche Ergebnis wie mit dem Befehl `at -l` erreichen wir auch mit ***atq*** (at-Queue).

```
#atq
```

```
4 2003-09-27 a root
```

Wenn wir einen Job wieder aus der Queue löschen wollen, verwenden wir dazu den Befehl ***atrm*** aus der Befehlsfamilie.

```
#atrm 4
```

Das war es, und unser Job mit der Nummer 4 wurde gelöscht. Die Eingabe von `atq` liefert keine Ausgabe mehr.

*Hinweis*

Der `at`-Befehl steht auch den Benutzern von Windows-Systemen zur Verfügung. Allerdings auf der Kommandozeile, die von den meisten Windows-Benutzern eher verschmäht wird. Schade eigentlich, denn viele Perlen liegen dort vergraben, und viele Sys-

temadministratoren von Windows-Systemen wissen auch, dass manche Aufgaben ohne Kommandozeile gar nicht lösbar wären.

Wenn wir die Jobs mit dem Kommando `at` in eine Queue zur Ausführung bereitstellen, wird diese Aufgabe zu der angegebenen Zeit durchgeführt, egal wie stark belastet das System gerade zu dieser Zeit ist. Wollen wir aber darauf Rücksicht nehmen, dass das System zu unterschiedlichen Zeiten unterschiedlich stark ausgelastet sein kann, dann verwenden wir anstelle von `at` den Befehl ***batch***. Der `batch`-Befehl arbeitet so, dass, wenn keine Zeit angegeben ist, der Job abgearbeitet wird, sobald die System-Ressourcen nur wenig ausgelastet sind. Wird eine Zeit angegeben, beginnt die Abarbeitung des Jobs, sobald die System-Ressourcen nach Ablauf dieser Zeit wieder gering ausgelastet sind. Die Auslastungsgrenze bedeutet, dass der ***Load*** einer Maschine unter 0.8 liegen muss. Die Optionen, die man in Verbindung mit `batch` verwenden kann, sind identisch mit denen vom `at`-Kommando.

Der SuperUser hat auf jeden Fall das Recht, diese Kommandos auszuführen. Die Rechte der anderen Benutzer werden über die Dateien ***/etc/at.allow*** bzw ***/etc/at.deny*** verwaltet. Wenn keine der beiden Dateien existiert, hat nur der SuperUser das Recht. In der Datei `/etc/at.allow` stehen die User mit Namen, die das Recht haben, die Befehle zu verwenden. Wenn diese nicht vorhanden ist, wird die Datei `/etc/at.deny` ausgewertet, und alle Benutzer, die nicht aufgelistet sind, dürfen die Befehle verwenden.

Was passiert aber, wenn unsere Reorganisation nicht nur einmal, sondern periodisch ablaufen soll? Hier hilft uns der `at` oder `batch`-Befehl nicht mehr weiter. Wir können uns aber mit dem ***cron*** (cronus)-Dienst oder besser Daemon aus der Affäre ziehen. Der ***crond*** (cron-daemon) startet definierte Programme und Befehle auf periodischer Basis. Der `crond` liest die Dateien ***/etc/crontab*** (cron Tabelle) aus. In dieser Datei stehen systemweite periodische Aufgaben. Des Weiteren liest der `crond` die Dateien in dem Verzeichnis ***/etc/cron.d/\**** aus. Alle diese Dateien besitzen das gleiche Format, auf das wir gleich genauer eingehen werden.

Der `crond` wacht jede Minute auf und überprüft all seine Einträge, die er über diese Dateien erhalten hat, ob irgendein Befehl zu dieser Minute auszuführen ist. Wenn ein Job abgearbeitet wird, werden alle Ausgaben des Jobs als E-Mail an den Besitzer

der crontab oder an jenen Benutzer, der in der **MAILTO**-Umgebungsvariablen hinterlegt ist, geschickt.

Der crond ist aber noch mächtiger. Er kann für jeden Benutzer des Systems eine eigene crontab-Datei verwalten. Mit dem Befehl **crontab** kann man diese Dateien verwalten. Die benutzerdefinierten Dateien werden unter dem jeweiligen Benutzernamen im Verzeichnis **/var/spool/cron/crontabs** gespeichert.

*Einschub*

Das Verzeichnis **/var/spool/cron** hat aber auch andere Einträge für **atjobs** und **atspool**. In atjobs stehen alle Aufgaben, die mit dem at-Befehl aktiviert wurden. Wenn wir uns mit cat so eine Datei ansehen, erkennen wir, dass es sich dabei eigentlich um ein Shell-Skript handelt, welches viele Shell-Variablen mit beinhaltet.

Der Befehl crontab verwaltet also die benutzerdefinierten crontabs.

**Tabelle 147: Optionen für crontab**

<b>Optionen</b>	<b>Bedeutung</b>
-u user	Die crontab des Benutzers user wird behandelt. Wenn diese Option nicht angegeben ist, dann wird die crontab des jeweiligen Users behandelt.
-l	Listet die crontab des Users auf dem Standard-Ausgabegerät auf.
-r	Die aktuelle crontab des Users wird gelöscht.
-e	Damit wird die crontab editiert. Und nur damit! Die crontab-Datei des jeweiligen Users wird mit dem in der Shell-Variablen EDITOR hinterlegten Editor geöffnet und kann angepasst werden.

In der crontab-Datei sind nun zeilenweise Einträge, die den Job, (den Befehl) der ausgeführt werden soll und die passende Zeit angeben. Der allgemeine Aufbau einer crontab-Zeitangabe ist wie folgt:

„Minuten“ „Stunden“ „Tag des Monats“ „Monat“ „Tag der Woche“

**Tabelle 148: crontab-Zeitbereiche**

<b>Feld</b>	<b>Werte</b>
Minuten	0-59
Stunden	0-23
Tag des Monats	1-31
Monat	1-12 (oder Namen)
Tag der Woche	0-7 (0 und 7 bedeuten Sonntag)

In jedem Feld ist auch ein \* als Wildcard möglich. Ein \* im Minutenfeld bedeutet, dass der Befehl jede Minute ausgeführt wird.

Hinter der Zeitangabe kommt der auszuführende Befehl, der in den allermeisten Fällen ein eigenes Shell-Skript ist.

MAILTO=heili

31 2 \* \* 0 cp /etc/passwd /backup/pass.bak

Dieser Eintrag in der crontab veranlasst den crond am Sonntag um 2:31 Uhr morgens, die Datei /etc/passwd nach /backup/pass.bak zu kopieren. Die Benachrichtigung bzw. die Ausgabe des Befehls wird dann dem Benutzer heili gemailt.

Jetzt soll natürlich nicht jeder den crond verwenden dürfen. Die Entscheidung darüber wird in den Dateien **/etc/cron.allow** und **/etc/cron.deny** getroffen. Wenn die Datei /etc/cron.allow existiert, muss der User, der den crond verwenden darf, darin aufgeführt werden. Wenn die Datei /etc/cron.allow nicht existiert, sondern nur die Datei /etc/cron.deny, dürfen alle User, die nicht aufgeführt sind, den crond verwenden. Wenn keine der beiden Dateien auf dem System-Vorhanden sind, darf ausschließlich der SuperUser diesen Dienst verwenden.



Mit der crontab-Datei können sehr komplexe Zeit- und Datumsangaben realisiert werden.

### 0 3 1 12 1 /go/reorg

Diese Zeile bedeutet, dass der Job reorg um 03:00 Uhr morgens ausgeführt wird, wenn der 1. Dezember ein Montag ist.

Es gibt nun ein weiteres System, das Aufgaben periodisch erledigen kann. Dieser Dienst heißt **anacron**. Mit dem anacron-Dienst kann man Befehle periodisch ausführen lassen mit einer Zeitangabe, die in Tagen definiert wird. Im Unterschied zum crond muss der Rechner dabei nicht 24 Stunden durchlaufen.

Wenn anacron gestartet wird, liest er normalerweise die Datei **/etc/anacrontab** ein. Diese Datei enthält eine Liste von Jobs, die vom anacron verwaltet werden. Diese Einträge beinhalten die Periode in Tagen, eine Verzögerung in Minuten, eine eindeutige Job-ID und das Shell-Kommando.

Periode    Verzögerung    Job-ID    Kommando

Die Fehler-Ausgaben, die der anacron erzeugt, werden zum syslogd geschickt. Die Ausgaben, die der Job erzeugt, werden gemailt.

**Tabelle 149: Optionen für anacron**

Optionen	Bedeutung
-f	Erzwingt die Jobausführung unabhängig vom Zeitstempel.
-u	Setzt den Zeitstempel der Jobs auf das aktuelle Datum, ohne aber irgendetwas auszuführen.
-s	Anacron startet den nächsten Befehl erst dann, wenn der vorherige beendet wurde.
-n	Startet die Jobs sofort, ohne auf eine mögliche Zeitverzögerung Rücksicht zu nehmen.

-d	Startet nicht im Hintergrund und gibt daher Fehlermeldungen auf der stderr und in den syslogd aus. Die Ausgabe des jobs wird wie gewöhnlich gemailt.
-q	Gibt keine Fehlermeldungen an die stderr aus (Nur zusammen mit -d möglich).
-t TAB	Verwendet TAB als anacrontab anstelle der Default-Datei.
-V	Versionsinformationen.

Im Unterschied zu den crontab's kann die anacrontab direkt editiert werden.

## 12.2

### Ein Muss - Sicherheitskopien

Eine der wohl wichtigsten Aufgaben eines Systemadministrators, egal aus welcher Systemwelt er kommt, ist es, eine passende Backupstrategie für das Unternehmen zu entwickeln. Das heißt, ein System zu definieren und zu planen, das bei diversen Fehlerursachen die möglichen resultierenden Standzeiten des Systems minimiert oder unter Umständen sogar ganz verhindert. Folgende Aspekte sollen bei einer Backupstrategie berücksichtigt werden. Dass diese Aspekte je nach Unternehmen unterschiedlich bewertet und gewichtet werden und sich dadurch die endgültige Backupstrategie entscheidend unterscheiden wird, ist offensichtlich.

1. Einteilung und Bedeutung von verschiedenen Daten eines Unternehmens. Die Daten müssen als Geschäftswert beurteilt werden, und die Kosten für eine Neugenerierung müssen betrachtet werden.
2. Die Datenintegrität muss gewährleistet werden. Änderungen, die während des Sicherns entstehen, müssen durch das Backup Berücksichtigung finden.

3. Speicherort - Backupziel. Das Backup von unternehmenskritischen Daten muss auf unterschiedliche Orte und Bereiche erfolgen (Disasterrecovery).
4. Backupzeitpunkt. Der eigentliche Durchführungszeitpunkt des Sicherungsprozesses sollte so gewählt werden, dass der produktive Betrieb nicht oder zumindest so wenig wie möglich davon beeinflusst bzw. behindert wird.
5. Netzwerkbackup. Die Möglichkeiten der Datenübertragung auf das Sicherungsmedium ist so zu wählen, dass ein Optimum an Datenübertragungsrate und Datensicherheit erreicht wird.
6. Zeitplan. Die Backupstrategie bedingt die Durchführung von Tasks, die täglich, wöchentlich oder monatlich laufen müssen.
7. Testwiederherstellung. Unabdingbar ist das Testen der Backupstrategie. Eine Testwiederherstellung auf Basis der angelegten Backups ist unbedingt durchzuführen, ansonsten verlässt man sich auf eine trügerische Sicherheit.

Nach systematischer Abarbeitung dieser 7 Aspekte können wir die Dateien und Verzeichnisse identifizieren, die mit unterschiedlichen Prioritäten gesichert werden sollen und müssen.

**Tabelle 150: Backuparten**

<b><i>Backup-Art</i></b>	<b><i>Beschreibung</i></b>
Vollständiges	Bei einem vollständigen Backup werden alle Dateien und Verzeichnisse, eben vollständig, auf das Backupmedium geschrieben.
Differentiell	Bei einem inkrementellen Backup werden nur die Dateien gesichert, die sich seit dem letzten vollständigen Backup geändert haben.
Inkrementell	Hier werden nur die Dateien gesichert, die sich seit dem letzten Backup geändert haben.

Kopie	Genau das, was das Wort ausdrückt. Eine einfache Kopie der Dateien oder Verzeichnisse.
Partiell	Ähnlich dem Vollbackup, aber es wird nicht das gesamte System gesichert, sondern nur ausgewählte Bereiche – Datenbereiche.

### Einschub

Wenn im Folgenden auf ein vollständiges Backup Bezug genommen wird, ist immer ein partielles Backup gemeint, da die unternehmenskritischen Daten in gewissen Teilen der Verzeichnishierarchie zu finden sind. Das System selbst wird meistens auch physikalisch auf eine andere Festplatte bzw. auf eine andere Partition installiert. Somit wird das eigentliche Betriebssystem oft einfach auf eine eingebaute Spareplatte gespiegelt. Damit reduziert sich die Ausfallszeit nach einem System-Headcrash auf einen Reboot.

Der Vorteil eines vollständigen Backups ist natürlich, dass alle Dateien auf einmal wieder zurückgespielt werden können. Allerdings bedingt ein vollständiges Backup unter Umständen sehr viel Platz und sehr viel Zeit, da immer alle Dateien gesichert werden. Die beiden anderen Arten bieten den Vorteil, dass weniger Speicherplatz verbraucht wird, und der eigentliche Backupprozess ist viel kürzer. Allerdings besteht der Nachteil, dass eine gewisse Reihenfolge beim Zurückspielen (**restore**) des Backups eingehalten werden muss.

Wir haben durch diesen kurzen Ausflug in das Organisatorische einen kleinen Einblick in die Komplexität eines Backups bekommen. Deshalb gibt es auch viele spezialisierte Programme, die ein komfortables und sicheres Backup von offenen Datenbanken und dergleichen ermöglichen. Dennoch bietet Linux schon eine ganze Reihe von „Bordmitteln“ an, mit denen man ein rudimentäres Backupsystem implementieren kann. Aufgrund der Komplexität der Aufgabe überrascht es uns nicht, dass auch die Befehle, die man dazu verwendet, sehr komplex und umfangreich sind.

Als Erstes kommen wir, wie versprochen, wieder auf den **tar**-Befehl zurück. Der tar-Befehl schreibt standardmäßig auf das

Default-Bandlaufwerk, das an der Maschine angeschlossen ist. Mit der Option `-f` kann man den Befehl auch veranlassen, anstelle des Bandlaufwerkes die angegebene Datei zu verwenden. Mit `tar` werden also alle Dateien und Verzeichnisse in eine einzige Archivdatei geschrieben. Das `tar`-Utility ist schon ein etwas älteres Baujahr, aber noch immer up to date und auf fast allen Systemen standardmäßig vorhanden.

Um ein Backup des Verzeichnisses `/home` in der Datei `/backup/home.tar` zu erstellen, verwenden wir folgendes Kommando:

```
#tar -cvfP /backup/home.tar /home/
```

Die Option `-P` bedeutet, dass die führenden `/` bei den Pfadangaben erhalten bleiben. Das bedeutet für Sie, dass die Daten nur an dem ursprünglichen Ort wieder hergestellt werden können.

**Tabelle 151: Optionen für den tar Befehl**

<b>Optionen</b>	<b>Bedeutung</b>
<code>-c</code>	Erzeugt ein Archiv.
<code>-x</code>	Entpackt ein Archiv.
<code>-t</code>	Listet den Archivinhalt auf.
<code>-f DATEI</code>	Stellt DATEI als Ein-Ausgabekanal anstelle des Defaultbandlaufwerkes <code>/dev/st0</code> ein.
<code>-A</code>	Hängt Daten an ein Archiv an.
<code>-d</code>	Findet Unterschiede (Differenzen) zwischen einem Archiv und einem Dateisystem.
<code>-r</code>	Hängt Dateien an ein Archiv an.
<code>-u</code>	Hängt Dateien nur an, wenn sie neuer sind als die im Archiv.

Für den `tar`-Befehl können aber auch noch zusätzlich folgende Optionen verwendet werden:

**Tabelle 152: Zusätzliche tat-Optionen**

<b>Optionen</b>	<b>Bedeutung</b>
--atime-re-serve	Verändert die Accesstime nicht.
-b N	Setzt die Blockgröße auf Nx512 Bytes (Default ist N=20).
-C DIR	Wechselt in das Verzeichnis DIR.
-F SCRIPT	Führt SCRIPT nach dem Ende jedes Bandes aus.
-G F	Erstellt, listet auf und extrahiert alte inkrementelle Backups nach GNU-Format.
-g F	Wie -G, nur im neuen GNU-Format.
-h	Anstelle von symbolischen Links werden die Dateien, auf die sie verweisen, gesichert.
-j	Komprimiert mit bzip2.
-k	Überschreibt alte Dateien nicht aus dem Archiv.
-L N	Wechselt das Band nach Nx1024 Bytes.
-M	Das Archiv kann sich auf mehrere Bänder verteilen.
-N DATE	Speichert nur Dateien, die jünger als DATE sind.
-p	Behält die Zugriffsrechte aus dem Archiv.
-P	Absolute Pfade werden gesichert.
--remove-files	Löscht die Dateien nach dem Hinzufügen zum Archiv.

-s same-owner	Extrahiert Dateien mit gleichem Eigentümer.
-T DATEI	Die Dateinamen, die erstellt oder gelesen werden sollen, kommen aus der Datei DATEI.
-v	Verbose-Ausgaben auf stdout.
-w	Interaktiv-, bei jeder Aktion wird nachgefragt.
-W	Nach dem Archivieren wird verglichen.
-X DATEI	Exkludiert Dateien aus dem File DATEI.
-Z	De-/Komprimiert mit compress.
-Z	De-/Komprimiert mit gzip.

```
#tar -cvf /backup/home.tar -g zeitstempel /home/
#tar -cvf /backup/home-inkr-1.tar -g zeitstempel /home/
```

Die erste Kommandozeile macht ein Vollbackup des Verzeichnisses /home/. Die zweite Kommandozeile erstellt ein inkrementelles Backup des gleichen Verzeichnisses. Wir erkennen, dass der Aufruf bis auf den Archivnamen gleich ist. In einem Shell-Skript kann man den Dateinamen der inkrementellen Backups z. B. mit dem aktuellen Datum zusammenstellen lassen. Versuchen Sie es gleich als Übung und als Wiederholung des Shell-Skriptings. (Als Hinweis: echo "Name-`date +%D`")

Ein weiteres Tool, das uns Archive erzeugt, ist **cpio** (copy in/copy out). Der Ausgabekanal von cpio wird mit dem >, < Umleitungsmechanismus gelenkt und kann daher ein normales File, eine Festplatte oder ein Bandlaufwerk sein. Als Eingabe wird eine Liste von zu kopierenden Dateien erwartet. Ein üblicher Weg, diese Liste zu generieren, ist der find-Befehl.

Das Tool `cpio` kennt somit drei Operationsmodi.

**Tabelle 153: Modi des `cpio` Befehls**

<b>Modus</b>	<b>Bedeutung</b>
<code>copy-in</code>	Kopiert Dateien aus einem Archiv heraus. Option <code>-i</code> .
<code>copy-out</code>	Kopiert Dateien in ein Archiv. Option <code>-o</code> .
<code>copy-pass</code>	Kopiert einen Verzeichnisbaum in einen anderen Verzeichnisbaum. Option <code>-p</code> .

Um bei unserem Beispiel von oben zu bleiben, machen wir ein vollständiges Backup des `/HOME`-Verzeichnisses.

```
#find /home | cpio -o > /backup/home.cpio
```

Mit der Option `-depth` beim `find`-Befehl kann man Fehler vermeiden, da zuerst die „Tiefe“ eines Verzeichnisses ermittelt wird und daraus heraus gearbeitet wird.

**Tabelle 154: Optionen für `cpio`**

<b>Optionen</b>	<b>Bedeutung</b>
<code>-a</code>	Setzt die Accesstime der Dateien nach dem Lesen wieder zurück.
<code>-A</code>	Fügt an ein bereits bestehendes Archiv an.
<code>-B</code>	Setzt die Blockgröße auf 5120 Byte. Default ist 512.
<code>--block_size=N</code>	Setzt die Blockgröße auf <code>Nx512</code> Byte.
<code>-c</code>	Verwendet das alte portable ASCII-Format.
<code>-d</code>	Erzeugt dort, wo es notwendig ist, die Verzeichnisse.



-I DATEI	Verwendet DATEI anstelle der Standard-Eingabe für die zu archivierenden Dateinamen.
-H FORMAT	Verwendet FORMAT als Archivformat: <ul style="list-style-type: none"> <li>• bin Überholtes Binary-Format.</li> <li>• crc Neues (SVR4) Format mit checksums.</li> <li>• newc Neues (SVR4) Format das mehr als 65536 i-Nodes unterstützt.</li> <li>• tar Altes tar-Format .</li> </ul>
-O	copy-out-Modus.
-i	copy-in-Modus.
-p	copy-pass-Modus.
-l	Verlinkt Dateien anstatt sie zu kopieren.
-L	Kopiert die eigentlichen Dateien, auf die der Link weist.
-m	Behält die Modifikationszeit bei.
--no-absolute-filenames	Kein absoluter Pfad im copy-in-Modus.
--no-preserve-owner	Ändert den Eigentümer beim copy-in und copy-pass-Modus nicht.
-r	Ändert interaktiv die Dateinamen.
-t	Liste des Inhalts eines Archivs.
-v	Verbose Ausgabe.

Mit dem `cpio`-Befehl kann man auch inkrementelle Backups erstellen. Allerdings ist dies eine Art work around mit dem `find`-Kommando.

```
#find /home/ -mtime -3 -depth | cpio -o > /backup/home.cpio.1
```

Überprüft man ein Archiv, kann folgende Kommandozeile werden:

```
#cpio -itvI /backup/home.cpio
```

Eingelesen in das aktuelle Verzeichnis wird das Archiv wieder mit:

```
#cpio -id < /backup/home.cpio
```

Das nächste Tool in diesem Themenkomplex ist der Befehl ***dd*** (disk dump oder direct dump). Das `dd`-Kommando wird dazu verwendet, um Dateien zu kopieren und dabei zu konvertieren.

**Tabelle 155: Optionen für `dd`**

<b>Optionen</b>	<b>Bedeutung</b>
<code>bs=BYTES</code>	Verwendet die angegebene BYTES Blocksize.
<code>cbs= BYTES</code>	Konvertiert BYTES auf einmal.
<code>if=DATEI</code>	Eingabedatei (oder Gerät).
<code>of=DATEI</code>	Ausgabedatei (oder Gerät).
<code>count=N</code>	Kopiert nur N Eingabeblocke.
<code>skip=N</code>	Es werden N Blocks übersprungen und dann erst kopiert.
<code>seek=N</code>	Es werden N Blocks in der Ausgabedatei übersprungen, bevor zu schreiben begonnen wird.
<code>obs=BYTES</code>	Es werden BYTES Bytes auf einmal geschrieben.

Das `dd`-Kommando kann für ein weites Spektrum von Aufgaben verwendet werden. Eine Aufgabenstellung, mit der man mit an

Sicherheit grenzender Wahrscheinlichkeit konfrontiert wird, ist ein Dualbootssystem zu installieren. Das heißt, Windows (NT,2k,XP) und Linux auf einer Maschine. Meine empfohlene Reihenfolge für die Installation einer Dualboot-Konfiguration ist: Zuerst Windows installieren. Den Windows-Bootmanager in den Master Boot Record der Festplatte schreiben lassen. Danach von CD booten und die Linux-Installation beginnen. Bei der LILO oder GRUB-Installation ist aber Vorsicht geboten. Sie haben die Festplatte partitioniert. Nun lassen Sie das Installationsprogramm den LILO oder GRUB auf die Partition schreiben, der Sie auch das ROOT-Verzeichnis zugewiesen haben. Nehmen wir einmal folgende Partitionen an:

/dev/hda1	----	Windows	NTFS
/dev/hda2	----	Linux	SWAP
/dev/hda3	----	Linux Nativ	/

Damit lassen wir den LILO oder GRUB in die Partition /dev/hda3 schreiben. Nun kommt ein entscheidender Moment. Wir wollen diese Bootinformationen dem Windowsbootmanager zugänglich machen. Deshalb wechseln wir, nachdem die Linux Bootinfos auf /dev/hda3 geschrieben wurden und noch bevor das System neu gestartet wird, mit ATL+STRG+Fx (x=1 bis 6) auf eine virtuelle Konsole. Dort erwartet uns irgendwo ein root-Prompt. Auf dieser Kommandozeile geben wir, nach Einlegen einer Floppy Disk, Folgendes ein:

```
#mkdir /floppy
#mount /dev/fd0 /floppy
#dd if=/dev/hda3 of=/floppy/bootsec.lin bs=512 count=1
```

Die Optionen beim dd-Befehl bedeuten, dass wir den Bootsektor der Partition hda3 einlesen und in die Datei bootsec.lin auf Diskette abspeichern wollen. Der Bootsektor ist 512 Byte groß. Damit wir aber nicht die gesamte Partition hda3 mit 512 Byte-Blöcken einlesen, beschränken wir es mit count=1, damit nur die ersten 512 Byte eingelesen werden. Damit haben wir den Bootsektor von Linux auf der Diskette in der Datei bootsec.lin.

Nun können wir das System neu starten lassen und booten zuerst wieder in das Windows-System. In Linux kommen wir ohnehin nicht, da wir dazu keine Informationen haben. Nachdem Windows gebootet ist, kopieren wir die Datei bootsec.lin von der Diskette auf die Platte c:\ und editieren die Datei

**c:\boot.ini.** Dort fügen wir die folgende Zeile am Ende der Datei ein:

```
C:\bootsec.lin="Betriebssystem Linux"
```

Abspeichern und fertig. Nun können wir beim Starten des Systems entscheiden, ob wir Linux oder Windows starten wollen.

Meine Empfehlung, den Windows Bootmanager zu verwenden gründet sich auf die Tatsache, dass es beim Einspielen von Windows-Service-Packs vorkommen kann, dass diese den Linux-Bootmanager überschreiben, weil sie erkennen, es handelt sich nicht um einen Standard Windows-Bootsektor und nehmen an, dass es sich um einen defekten Bootsektor handelt und reparieren ihn. Dies kommt zwar in letzter Zeit nicht mehr vor, aber kann nicht gänzlich ausgeschlossen werden. Deshalb diese Empfehlung, die eine rein persönliche Einstellung widerspiegelt.

Letztendlich muss man beim Themenkomplex Backup die Befehlskombination **dump** und **restore** erwähnen.

Der Befehl dump ist ein ext2-Backuptool. dump durchsucht Dateien auf einem ext2-Dateisystem und findet heraus, welche Dateien gesichert werden müssen. Diese Dateien werden auf die angegebene Harddisk, Bandlaufwerk oder Datei geschrieben. Ein dump, der größer als das Ausgabemedium ist, wird auf mehrere Medien aufgeteilt. Normalerweise wird die Größe eines Mediums durch ein **end-of-media**-Signal ermittelt. Wenn allerdings Geräte (Medien) verwendet werden, die nicht in der Lage sind, solch ein Signal zu generieren, wird jedes Medium mit einer fixen Größe behandelt. Dies ist oft bei Bandlaufwerken der Fall.

Die Dateien, die zu dumpen sind, sind entweder ganze Dateisysteme oder eine Liste von Dateien und Verzeichnissen.

**Tabelle 156: Optionen für dump**

<b>Optionen</b>	<b>Bedeutung</b>
-0 bis 9	Dumplevel. Vollbackup = 0. Höhere Ziffern bedeuten ein inkrementelles Backup von Dateien
-a	Automatische Größe. Ignoriert alle Größenangaben und schreibt bis zu einem end-of-media.

-A DATEI	Schreibt eine Inhaltsangabe in die Datei DATEI. Diese wird dann von restore verwendet.
-b SIZE	Setzt die Blockgröße. Default ist 10
-c	Ändert die Einstellungen für ein Bandlaufwerk mit der Dichte von 8000bpi (Bytes per Inch) und einer Länge von 1700 Fuß. Wenn man ein Bandlaufwerk angibt, wird end-of-media-Erkennung deaktiviert.
-d DICHTe	Setzt die Dichte eines Bandes. Default ist 1600bpi.
-f DATEI	Schreibt das Backup in das File DATEI.
-j COMP_LEV	Komprimiert mit bzlib. Der Kompressionsgrad ist COMP_LEV.
-L LABEL	Setzt einen Medienlabel.
-M	Aktiviert den Multi-Volume-Modus.
-z COMP_LEV	Komprimiert mit zlib. Der Kompressionsgrad ist COMP_LEV.
-W	Dem Administrator wird mitgeteilt, welche Dateisysteme zum Sichern sind.
-u	Updatet die Datei /var/lib/dumpdates. Hierin werden Einträge gemacht, die den Dateisystemnamen, Inkrement-Level und ctime angeben.

Man sollte bei der Verwendung von dump immer zuerst mit einem Level 0-Backup beginnen.

```
#dump -0u -f /dev/st0 /usr/src
```

Mit dem Tool **restore** kann man Backups, die mit dump gemacht worden sind, wieder zurückspielen.

**Tabelle 157: Optionen für restore**

<b>Optionen</b>	<b>Bedeutung</b>
-i	Erlaubt das interaktive Zurückspielen von Dateien.
-R	Es wird nach einem spezifischen Band von einem Multi-Volume-dump gefragt.
-r	Rebuild eines ganzen Dateisystems.
-t	Angegebene Dateinamen werden gelistet, wenn sie im Backup enthalten sind.
-x	Extrahiert die angegebenen Dateien und Verzeichnisse.

**Tabelle 158: Zusätzliche Parameter für restore**

<b>Parameter</b>	<b>Bedeutung</b>
-A DATEI	Liest den Inhalt des Archivs von der Datei DATEI.
-f DATEI	Liest das Backup von DATEI ein.
-h	Extrahiert in das aktuelle Verzeichnis.
-M	Aktiviert den Multi-Volume-Modus.
-V	Aktiviert das Lesen von Multi-Volumes, die keine Bandlaufwerke sind, z. B. CD-ROMS.
-y	Beantwortet alle Fragen automatisch mit Ja..

```
#restore rf /dev/st0
```

Dieser Befehl stellt eine ganze Partition wieder her.

```
#restore if /dev/st0
```

Dieser stellt einzelne Dateien mit interaktivem Eingriff wieder her.

Auch wenn die Leistungen von Linux bisher schon äußerst beeindruckend waren, blüht sie erst so richtig auf, wenn auch noch die Netzwerkfähigkeiten betrachtet werden. Linux ist das Netzwerk. Zwei Drittel aller Rechner im Internet sind Linux basierend. Der Leitspruch von Sun Microsystems bringt es auf den Punkt: „*We are the Dot in .com*“. Wir werden in diesem Kapitel alles Notwendige über die Netzwerktechnik lernen, um damit arbeiten zu können. Allerdings ersetzt dieses Kapitel bei weitem nicht das Studium anderer einschlägiger Literatur, da es sich um eine sehr komplexe Materie handelt.

Wir werden hier die wichtigsten Protokolle und Dienste sowie deren Konfiguration besprechen und auch einen kurzen Blick in die geschichtliche Entwicklung werfen.

### 13.1

#### Eine kurze Geschichte

Die Wurzeln des heutigen Internets und damit auch der Standardprotokolltypen, die die heutigen modernen Netzwerke dominieren, liegen in der Zeit des kalten Krieges. Die massive Angst der Amerikaner, einem russischen Erstschatz nicht entsprechend antworten zu können, war auch Anlass für ein Aufrüsten der Informationstechnologie. Man erkannte sehr bald, dass die damals zur Verfügung stehenden Mittel (vernetzte Computer) nicht ausreichend waren, denn fiel nur ein Rechner und damit ein Netzwerkknoten aus, stand alles. Das **DoD** (Department of Defense) gab nun den Startschuss für die Entwicklung eines ausfallsicheren Netzwerkes. Damit war **ARPA** (Advanced Research Project Agency) und schließlich das **ARPA-Net** geboren. Das ARPA-Net hat 1969 16 Standorte miteinander verbunden. Das war der Anfang der paketvermittelnden Netzwerkarchitekturen. In den kommenden Jahren wurden das Verfahren und die Techniken immer weiter verfeinert, bis dann endlich 1978 die Protokolle **TCP**, **UDP** und **IP** in der im Wesentlichen noch heute gültigen Fassung vorlagen.

1979 haben zwei Studenten zwei Unix-Rechner mittels Telefonleitung miteinander verbunden und verwendeten zum Datenaustausch **UUCP** (Unix to Unix Copy Protocol). Das System arbeitete so, dass ein Rechner eine Nachricht auf einen Server kopiert, die dieser zwischenspeichert. Von dort wird die Nachricht vom zweiten Rechner abgefragt. Dieses System existiert noch heute unter dem Namen **UseNet** oder **NewsGroups**.

1982 wurde **TCP/IP** zum Standard erhoben und damit der Startschuss für die Umrüstung des ARPA-Nets gegeben. Im gleichen Jahr wurde in Europa das **EUnet** installiert. Ein weiterer Standard, der heute auch der meistgebrauchte im Internet ist, wurde ebenfalls 1982 verabschiedet – die **E-Mail**.

Der TCP/IP-Code wurde letztendlich in das Berkely-Unix aufgenommen und erhielt dadurch rasche Verbreitung. MS-DOS-Rechner erhielten über das FidoNet Zugang zum sich entwickelnden Internet.

1984 wurde die „magische“ Grenze von 1000 vernetzten Rechnern überschritten und damit der Wunsch nach einem schlagkräftigen System der Namensverwaltung immer größer. Dies war die Geburtsstunde des **DNS** (Domain Name Service)-Dienstes.

Mit der Implementierung von TCP/IP in Windows und der steigenden Leistungsfähigkeit der Personal Computer konnte der TCP/IP-Protokollstack seinen Siegeszug in der Netzwerkwelt, trotz des großen Overheads, antreten und eigentlich vollenden. Heutige Netzwerke arbeiten fast ausschließlich mit dieser Protokoll-Suite.

## 13.2

### Schichten und Netzwerktypen

Im Laufe der Zeit hat sich TCP/IP als die Sprache, die Computer in einem Netzwerk untereinander sprechen, um sich zu verständigen, etabliert. Dennoch ist es nicht alleine auf der Welt. Die Einteilung der Protokolle geschieht in sogenannten Schichten. Diese Schichten wurden von der **ISO** (International Standardization Organisation) definiert und standardisiert. Das heißt, wer sich bei seinen Entwicklungen an diesen Schichten orientiert, wird kompatibel sein. Das ISO hat das **OSI** (Open System Interconnection) -Schichten-Modell folgendermaßen definiert:



**Tabelle 159: ISO/OSI-Schichten**

<b>Schicht</b>	<b>Bedeutung</b>
1. Bitübertragung oder Physical layer	Übertragung des Bitstroms über den Übertragungskanal.
2. Sicherung oder Data link layer	Paketierung des Bitstroms mit Übertragungsfehlererkennung.
3. Vermittlung oder network layer	Bestimmung des Übertragungswegs.
4. Transport oder transport layer	Bereitstellung einer Ende-zu-Ende-Verbindung.
5. Kommunikation oder session layer	Behandlung von sitzungsrelevanten Ereignissen wie Verbindungsabbruch.
6. Darstellung oder presentation layer	Umwandlung der Daten in ein einheitliches Format.
7. Anwendung oder application layer	Protokolle der Anwendung wie z. B. E-Mail.

Die Schichten 1 bis 3 sind für die Kommunikation auf der Übertragungsstrecke zuständig. Die Schichten 4 bis 7 werden von Sender und Empfänger verwendet.

Wenn man von TCP/IP spricht, meint man eine ganze Protokoll-Suite. Allerdings passt diese nicht korrekt in das Sieben-Schichten-Modell hinein, da die Schichten 1 und 2 sowie 4 bis 7 zusammenfallen.

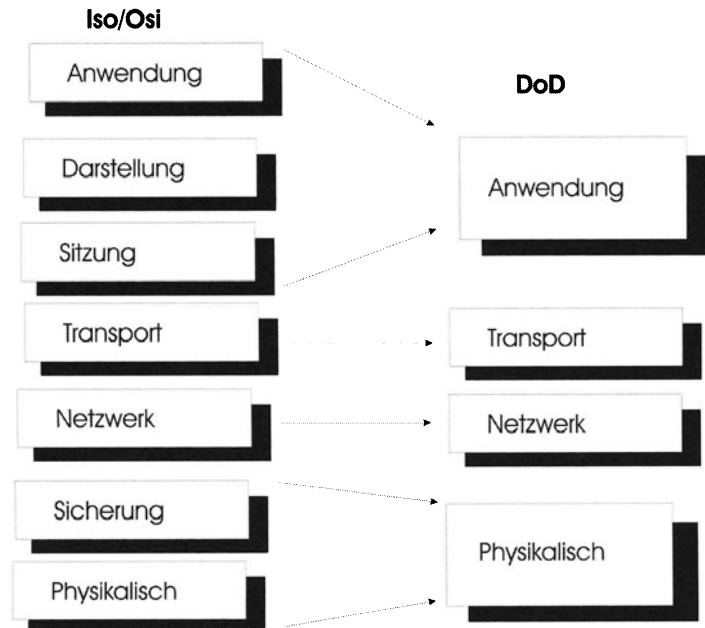


Abbildung 26: Gegenüberstellung DoD und ISO/OSI

Des Weiteren finden wir zwei Kommunikationswege, eine vertikale Kommunikation innerhalb eines Rechners und eine horizontale zwischen den Rechnern. Daher ist die Definition von Schichten sinnvoll, weil man definierte Schnittstellen, also Dienste von einer Schicht erwartet und dies anbieten muss, wenn man z. B. einen neuen Netzwerkkarten-Treiber programmiert.

Als Transporttechnologie hat sich im Großen und Ganzen die Ethernettechnologie durchgesetzt. Ab und an findet man auch noch Token Ring als Netzwerkarchitektur.

Das Ethernet wird vom IEEE unter der Nummer **802.3** standardisiert und ist sicherlich die vorherrschende Technologie, die bei Netzwerken gefunden werden kann. Deshalb möchte ich hier auch etwas näher darauf eingehen.

Ursprünglich hatte man das sogenannte **Thick Ethernet** (10Base5). Dies ist ein Koaxialkabel mit einem Durchmesser von 1,27 cm und meistens mit der Farbe gelb. Zum Anschluss an einen Computer ist ein sogenannter **Transceiver** notwendig. Bei diesem Typ handelt es sich um eine Bus-Verkabelung, die, um Reflexionen der Signale zu vermeiden, an beiden Enden mit ei-

nem 50 Ohm-Widerstand abgeschlossen werden muss. Wir erinnern uns an den Bus-Terminator von SCSI.

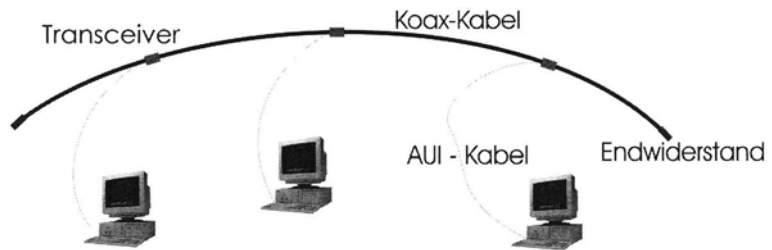


Abbildung 27: Thick-Ethernet

Der Durchbruch, und damit auch die große Verbreitung des Ethernet, kam mit dem sogenannten **Cheapernet** oder **Thin Ethernet**. Das Kabel wird ebenfalls als Bus verlegt und die Computer in Serie mittels T-Stücke angeschlossen. Der Transceiver ist nicht mehr notwendig. Das Kabel ist immer noch ein Koax-Kabel (RG-58), allerdings viel dünner und handlicher. Auch hier muss der Bus mit 50 Ohm-Widerständen abgeschlossen werden. Die maximale Geschwindigkeit, die mit diesem Ethernet Typ erreicht werden kann, ist mit 10Mbit/s begrenzt.

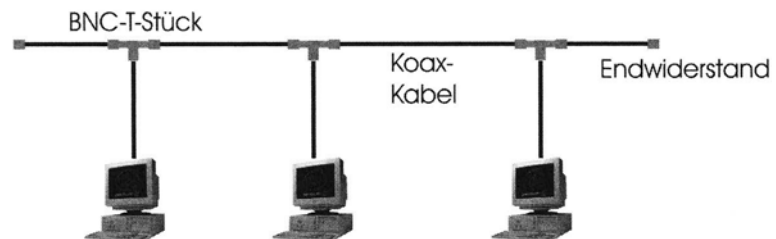
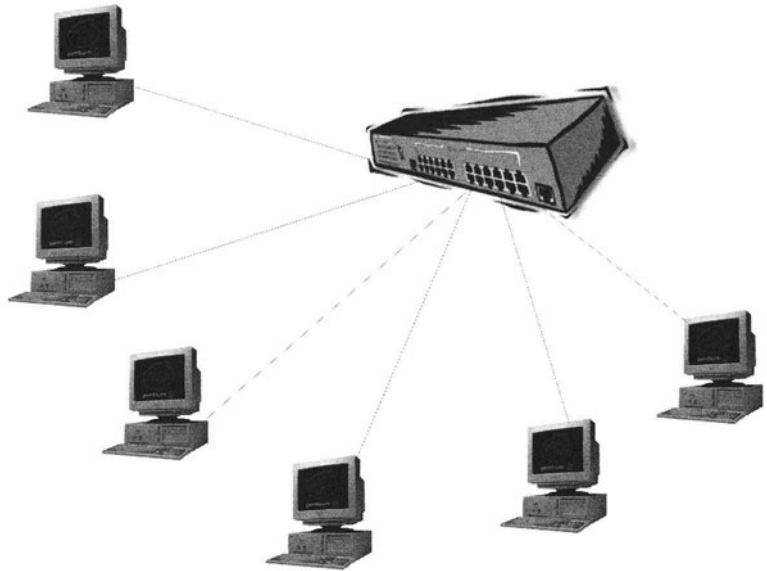


Abbildung 28: Thin Ethernet RGB-58

Mit der wachsenden Komplexität der Vernetzungsbedürfnisse wurden bald die Grenzen von Thin Ethernet bezüglich Geschwindigkeit, Begrenzung der maximalen Ausdehnung und die Problematik der Skalierbarkeit immer enger. Daraus entstand die 10BaseT-Technologie. Hier ist das zentrale Merkmal die Sternverkabelung über einen Hub oder Switch. Der Anschlussstecker ist ein RJ-45-Stecker, so wie er bei ISDN verwendet wird. Mit

10BaseT ist man von der Geschwindigkeit allerdings noch immer eingeschränkt. Erst mit 100BaseT wird die Geschwindigkeit auf 100Mbit/s gesteigert. Derzeit mausert sich allerdings schon 1000BaseT schön langsam zum Standard. Die Servercomputer, die man derzeit kaufen kann, haben alle schon Netzwerkkarten, die 10/100/1000Mbit/s mit Autoerkennung unterstützen.



*Abbildung 29: 10Base-T (Sternverkabelung)*

Die Glasfaserverkabelung gewinnt auch immer mehr an Bedeutung. So findet man in der Gebäudeverkabelung immer häufiger Glasfasertechnologien **FDDI** (Fiber Distributed Data Interconnect). Hier werden die Daten nicht mehr elektrisch, sondern mit Licht übertragen. Der Vorteil ist offensichtlich – es sind keine elektromagnetischen Störungen möglich und damit auch größere Ausdehnungen der Netze realisierbar. Ein FDDI-Netzwerk ist eigentlich ein Token Ring-Netzwerk mit einer Übertragungsgeschwindigkeit von 100Mbit/s. Die Ring-Struktur wird durch jeden Rechner gezogen. Fällt ein Rechner im Netzwerk aus, schließen die Nachbarrechner automatisch den Ring wieder, und das Netzwerk bleibt funktionstüchtig.

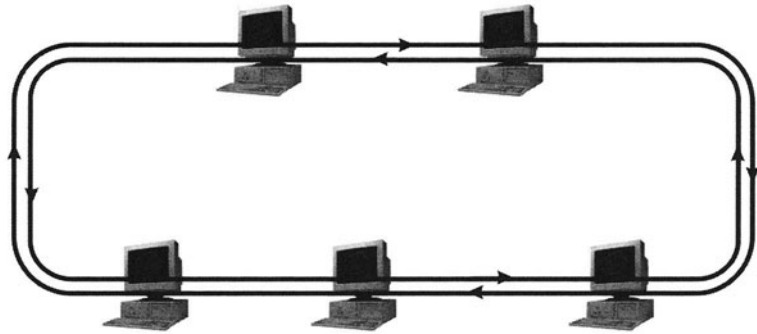


Abbildung 30: FDDI

Zum Abschluss noch ein Blick auf **Token Ring**. Token Ring wurde von IBM entwickelt und war schon früh in großen Unternehmen vertreten. Technologisch ist Token Ring ein Vertreter der Spitzenklasse. Aufgrund des höheren Preises konnte sich Token Ring trotz oft besserer technologischer Parameter nicht so stark durchsetzen bzw. die Verbreitung blieb weit hinter der von Ethernet zurück. Das Markenzeichen von Token Ring ist der **Token** und die Ring-Struktur der Verkabelung.

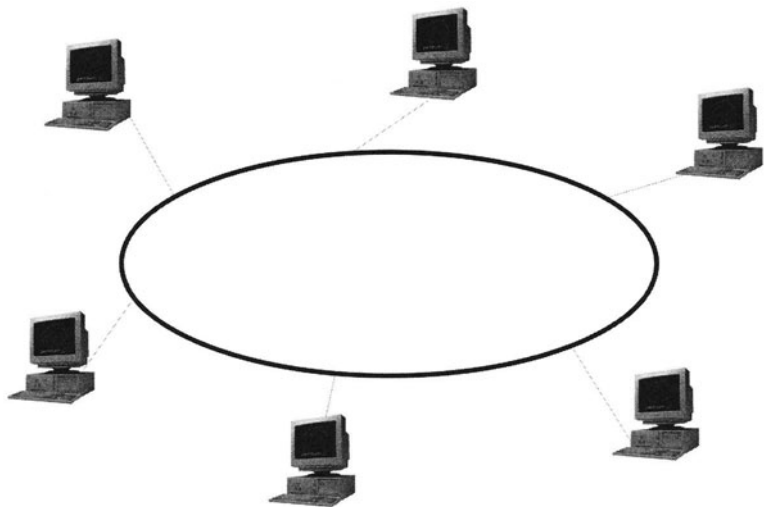


Abbildung: 31: Token Ring

Der Token läuft ständig im Ring in eine Richtung, und auf diesen Token werden die Daten für die Übertragung aufgeladen und

beim Empfänger heruntergenommen. Da jede Station den Token einliest und wieder ausgibt, ist die Übertragungsqualität gegenüber Ethernet wesentlich besser. Beim Ethernet sendet jede Station wann sie will. So kann es passieren – und es passiert sehr oft – dass zwei oder mehrere Stationen zur selben Zeit senden wollen. Es kommt zu einer sogenannten Kollision. Danach bleibt das Netzwerk kurze Zeit stehen und beginnt dann alles wieder von Neuem. Jede sendewillige Station wartet einige Microsekunden (Zufallswert) und versucht dann neuerlich zu senden. Damit ist klar einzusehen, dass die Anzahl der Stationen in einem Ethernet begrenzt ist. Ein Ethernet in dieser Form wird auch **Collision Domain** genannt. Das Übertragungsverfahren heißt **CSMA/CD** (Carrier Sense Multiple Acces / Collision Detection).

**Tabelle 160: Netzwerktypen**

<i><b>Typ</b></i>	<i><b>Beschreibung</b></i>
LAN	Local Area Network. Haben normalerweise eine Ausdehnung von wenigen Kilometern innerhalb eines Gebäudes oder zwischen Gebäuden (Campus).
MAN	Metropolitan Area Network. Ausdehnung von einigen zig Kilometern. Werden als Kopplung von LANs und Telekommunikationsdiensten realisiert.
WAN	Wide Area Network. Landesweite und kontinentale Netzwerke. Kopplung von LAN, MAN und Telekommunikationsdiensten.
GAN	Global Area Network. Weltumspannendes Netzwerk. Kopplung von LAN, MAN, WAN und Telekommunikationsdiensten.

## 13.3

**Protokolle, Adressen und Ports**

TCP/IP ist das Synonym für eine ganze Protokollfamilie. Betrachten wir einmal die wichtigsten aus dieser Familie:

**Tabelle 161: Protokolle**

<b>Protokoll</b>	<b>Beschreibung</b>
IP	Internet Protocol. Protokoll der Schicht 3 – Vermittlung. Der wichtigste Parameter, den uns IP gibt, ist die IP-Adresse. Die Adresse eines Rechners in einem TCP/IP-Netzwerk. Verbindungslos.
TCP	Transmission Control Protocol. Protokoll der Schicht 4. Es sorgt für einen reibungslosen Transport von Sender zu Empfänger. Teilt den Datenstrom in übertragbare Teile auf. Verbindungsorientiert.
UDP	User Datagramm Protocol. Protokoll der Schicht 4. UDP ist für schnelle Datenübertragung optimiert. Verbindungslos.
ICMP	Internet Control Message Protocol. Protokoll der Schicht 3. Fehler- und Kontrollnachrichten, die IP betreffen, werden übertragen.
ARP	Adress Resolution Protocol. Protokoll der Schicht 2. Findet zu einer gegebenen IP-Adresse eine MAC (Media Access Control) oder Hardware-Adresse.

Bevor wir das noch so kleinste Bit übertragen können, müssen wir das Netzwerk auf unserem System einrichten. Das bedeutet, alle notwendigen Parameter zu konfigurieren, die man für den Betrieb des Netzwerkes benötigt.

Am Anfang steht die IP-Adresse. Die derzeitige IP-Implementation ist die Version 4. Aufgrund des Adressenmangels weltweit

wird eine Umstellung auf die Version 6 diskutiert. Nachdem der Druck durch das sogenannte **NAT** (Network Address Translation) etwas aus der Diskussion genommen wurde, wird die Umstellung noch etwas auf sich warten lassen. Mit NAT kann man einfach viele private Adressen hinter wenigen offiziellen „verstecken“, man benötigt somit weniger IP-Adressen für die gleichen Zwecke.

Die IP-Adresse ist eine eindeutige Netzwerkadresse für einen Rechner. Eine IP-Adresse kann ohne eine Netzwerkmaske nicht alleine existieren, denn erst dann kann man eine Zugehörigkeit zu einem bestimmten Netzwerk eindeutig klären.

Eine IP-Adresse setzt sich aus vier 8 Bit-Zahlen zusammen und hat daher eine gesamte Länge von 32 Bit.

Das heißt, die Zahlen können zwischen 0 und 255 sein. Dass dem nicht ganz so ist, werden wir weiter hinten sehen.

Im Kopf (Header) des IP-Paketes stehen die IP-Adressen. Es finden sich dort die Absender IP-Adresse oder Source IP-Adresse genannt und die Ziel IP-Adresse oder Destination IP-Adresse. Die IP-Adresse wird in zwei Teile unterschieden. Ein Teil ist die eigentliche Host (Rechner oder Node)-Adresse und der zweite Teil ist der Netzwerkteil, an dem die Hostadresse partizipiert. Die Zuweisung von weltweit eindeutigen IP-Adressen erfolgt vom **NIC** (Network Information Center) des jeweiligen Landes.

Es können zwei Arten von Adressen unterschieden werden. Die klassenbehafteten Adressen und die in letzter Zeit immer wichtiger werdenden klassenlosen Adressen.

**Tabelle 162: Netzwerkklassen**

<b>Klasse</b>	<b>Klassenbits</b>	<b>Aufteilung</b>
A	0	Netz (7)    Host (24)
B	10	Netz (14)    Host (16)
C	110	Netz (21)    Host (8)
D	1110	Multicast Adresse
E	11110	Reserviert



Die Netzwerkadressen mit lauter Einsern oder lauter Nullern haben besondere Bedeutung und stehen somit nicht zur Adressierung zur Verfügung.

Da wir diese Adresse auf Binärebene behandeln können, können wir uns auch sofort ausrechnen, wieviele Hosts in wievielen Netzen man adressieren kann.

Nehmen wir einmal eine Klasse-A-Adresse her. Hier kann man  $2^7-2=126$  Netzwerke und  $2^{24}-2=16.777.214$  Rechneradressen vergeben. Die minus 2 kommen von 0 und 255. Das heißt, die oberste und die unterste Adresse fällt immer weg, da sie besondere Bedeutungen haben.

### Übung

Versuchen Sie, für die anderen Netze die Netz- und die Hostanzahl zu errechnen.

Netzwerkadressen werden natürlich nicht im Binärformat geschrieben, sondern im gepunkteten Dezimalformat (dotted decimal).

192.168.0.1 lautet in der binären Schreibweise:

1100000.10101000.00000000.00000001

Die ersten drei Bytes oder auch Octetts genannt sind der Netzwerkteil, und die restlichen 8 Bit sind der Hostteil. Es handelt sich hierbei um eine Klasse C-Adresse. Die Klasseneinteilung ergibt sich somit zu:

A: 1 - 127

B: 128 - 191

C: 192 - 223

Wenn wir also eine C-Adresse verwenden, dann haben wir zwei Adressen weggenommen. Nehmen wir wieder unsere C-Adresse von vorhin 192.168.0.1. Diesem Adressraum einer C-Klasse fehlen zwei Adressen und zwar die Adresse 192.168.0.0 und 192.168.0.255. Die erste Adresse definiert das aktuelle Netzwerk. Ist also die Definition des Netzwerkes, in dem die Rechner eine Adresse zugewiesen bekommen. Die zweite Adresse ist die sogenannte Broadcast-Adresse. Das bedeutet, wenn so eine Adresse auf dem Netzwerk auftaucht, fühlt sich jeder Rechner davon angesprochen.

Das IP-System kennt nun noch das Konzept des Subnettings. Das heißt, man teilt ein bestimmtes Netzwerk in weitere Teilnetze auf. Dazu benötigt man aber eine zusätzliche Komponente, die Netzwerkmaske oder **Subnetmaske** genannt wird. Bei der Behandlung von Netzwerk-Adressen werden die IP-Adresse und

die Subnetmask „verUNDet“ also miteinander binär UND verknüpft. Das bedeutet:

```

UND 0 1
    0 0 0
    1 0 1

```

Wenn wir eine Klasse C-Adresse 192.168.0.1 vollständig erhalten, also nicht weiter unterteilen wollen, ist die Netzwerkmaske gleich 255.255.255.0. Denn jede 1 in der Maske verändert den Netzteil nicht, und die Hostadresse bleibt wie sie war.

```
192.168.0.1    11000000.10101000.00000000.00000001
```

```
255.255.255.0  11111111.11111111.11111111.00000000
```

```
Ergebnis:     11000000.10101000.00000000.00000001
```

Wir erhalten also einfach wieder unsere Ausgangs-IP-Adresse.

Der Netzwerkteil wird von links her geschrieben. Wenn wir unsere C-Adresse weiter in Netze unterteilen wollen, erweitern wir den Netzteil einfach um so viele Bits wie wir Netze benötigen. Mit einer Binärzahl kann man genau zwei Zustände definieren, 0 und 1. Damit kann man aber, mit einem Bit mehr in der Netzwerkmaske, zwei zusätzliche Netze beschreiben. Wollen wir 4 zusätzliche Netze definieren, müssen wir um zwei Bits verlängern.



Abbildung 32: IP-Adressen Aufbau

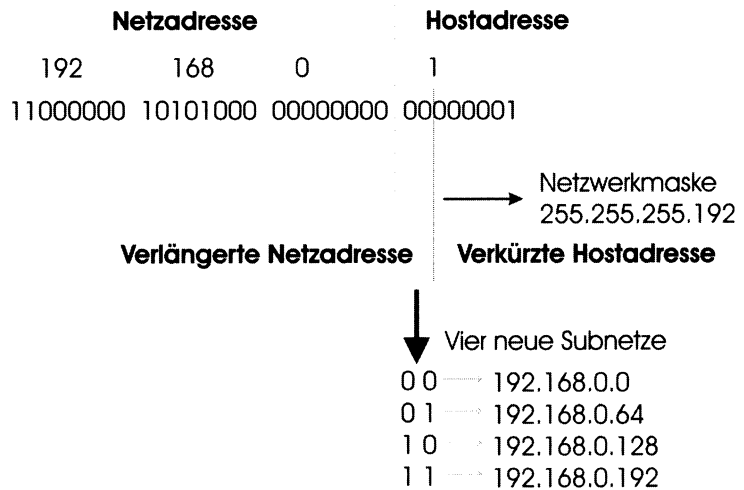


Abbildung 33: Subnet

Wenn wir allerdings die Netzmaske um ein Bit oder zwei Bit verlängern, muss sich auch deren Wert ändern. Wenn wir wieder die Wertigkeiten 128 64 32 16 8 4 2 1 hinterlegen und von links kommen, ist der Wert des 4. Bytes der Netzmaske bei einem Bit 128 und bei zwei Bits 192.

Erweitern wir um 2 Bits, wir generieren also 4 zusätzliche Subnetze. Die Netzwerkmaske verändert sich zu 255.255.255.192. Mit den 2 Bits können wir 4 Zustände darstellen, und diese definieren uns die Teilnetze:

1. Teilnetz 192.168.0.0
2. Teilnetz 192.168.0.64
3. Teilnetz 192.168.0.128
4. Teilnetz 192.168.0.192
- 5.

Die dazugehörigen Broadcast-Adressen ergeben sich wie folgt:

1. Teilnetz 192.168.0.63
2. Teilnetz 192.168.0.127
3. Teilnetz 192.168.0.191
4. Teilnetz 192.168.0.255

Der Computer unterscheidet bzw. entscheidet nun aufgrund des Ergebnisses der VerUNDung der eigener IP-Adresse mit der Subnetmaske und der fremden IP-Adresse mit der Subnetmaske, ob die Destination IP-Adresse im gleichen Netzwerkteil ist oder nicht. Und damit, ob sie bzw. der Rechner direkt über das LAN erreichbar ist oder nicht. Der Entscheidungsprozess läuft so, dass, wenn das Ergebnis der beiden binären UND-Operationen gleich ist, der Zielrechner im gleichen Subnetz zu finden ist.

Wenn nicht, was dann? Dann kommt eine weitere Komponente des Netzwerks zum Einsatz, die Default-Route. Also der Standardweg. Die IP-Schicht ist für die Wegewahl, also die Vermittlung von Senderechner zu Empfangsrechner, zuständig. Dieser Weg wird **route** genannt. Nachdem der Rechner selbst nur das eigene Netzwerk kennt, gibt man ihm einen Rechner oder besser gesagt Router an, an den alle Computer im Netzwerk die Pakete schicken sollen, deren Zieladresse nicht im eigenen Netzwerk liegt. Dieser Weg heißt Default-Route. Oft wird man auch den Begriff Gateway dafür hören. Die Default-Route ist also ein IP-Adressen-Eintrag für einen Rechner im eigenen Netz. Hat man mehrere Router bzw. Wege aus dem eigenen Netzwerk hinaus in die Welt, kann man für gewisse Ziel-Adressen spezielle Wege angeben. Diese eigens definierten Wege heißen **statische Routen**, da man sie statisch, also fest vorgibt. Alle Zieladressen, die nicht von statischen Routen abgedeckt werden, werden zum Default-Router geschickt. Dieser kennt den weiteren Weg. Die Default-Route ist meistens die vom Internet-Provider zugewiesene Routeradresse.

### *Einschub*

Die vorhin schon angesprochene Verknappung der IP-Adressen hat dazu geführt, dass das klassische Klassenmodell durch ein klassenfreies adaptiert wurde. Dabei werden Adressbereiche geografischen Zonen zugeordnet. Z. B.:

Europa                      194.0.0.0 bis 195.255.255.255

Nordamerika              198.0.0.0 bis 199.255.255.255

Die Größe der jeweiligen Bereiche muss jetzt nicht mehr mit der Netzgröße der jeweiligen Klasse identisch sein. So kann ein Netz im Klasse C-Bereich gewählt werden, der größer als diese ist.

Das heißt, die Netzwerkmaske alleine definiert die Größe des Netzwerkes. Diese Art nennt man **CIDR** (Classless Inter Domain Routing). Allerdings unterstützen nicht alle Router CIDR.

Wir haben die Schreibweise der IP-Adressen als dotted decimal kennen gelernt. Oft findet man aber die Angabe der IP-Adresse

gemeinsam mit der Netzmaske in der Form  $x.x.x.x/y$ , wobei  $y$  die Anzahl der gesetzten Netzmaskenbits angibt. Das bedeutet, eine IP-Adresse 192.168.0.1 mit Netzmaske 255.255.255.0 kann man auch in der Form 192.168.0.1/24 schreiben und das zugehörige Netz dann so benennen 192.168.0.0/24.

Wenn wir bei dieser Adresse nun ein Subnetting mit 4 zusätzlichen Netzwerken (also 2 Bits länger) haben, dann ändert sich die Angabe von 192.168.0.1 mit 255.255.255.192 zu 192.168.0.1/26.

Nachdem wir nun wissen wohin mit unseren Paketen – wir kennen den Weg – soll auch der Transport beginnen. Dafür ist das **TCP** (Transmission Control Protocol) zuständig. In dessen Headerinformationen finden wir auch die Angaben über die zu verwendenden Ports (Ziel-Port oder Destination-Port und Source-Port oder Absende-Port). Der Port ist jene Eingangstür der angesprochenen Rechner, wo der jeweilige Serverdienst horcht und der Client Kontakt aufnehmen will. Jeder Service – jeder Serverdienst – hat über die Zeit einen immer wieder dafür verwendeten Port als Standardport zugewiesen bekommen. So hört ein Web-Server standardmäßig auf den Port 80. Wir können mit einem Web-Server nur dann Kontakt aufnehmen, wenn wir neben der IP-Adresse des Servers auch den Port 80 mit angeben.

Sie werden jetzt sicher sagen, dass Sie dies in ihrem Browser noch nie getan haben. Das stimmt nur indirekt, denn der Web-Browser ist eine spezialisierte Anwendung und hängt automatisch den Port 80 bei einer Anfrage an. Damit der anfragende Computer auch Daten empfangen kann, muss auch dieser einen Port, also eine Eingangstür öffnen. Diese wird im TCP-Header mit angegeben. Damit weiß auch der entfernte Rechner, wohin er die Datenpakete zurückschicken kann.

Jeder Dienst hat somit einen Standardport. Man spricht auch von den **well known ports**.

**Tabelle 163: Port-Nummern**

<b>Port</b>	<b>Dienst</b>
22	Secure-Shell
23	Telnet

25	E-Mail (SMTP Simple Mail Transfer Protocol)
53	Domain Name Service
80	Web-Server - http (Hypertext Trasfer Protocol)
109	Post office Protocol (POP) Version 2
110	POP3
119	News-Dienst (NNTP News Network Transfer Protocol) - UseNet
139	Windows NetBIOS
143	IMAP- Internet Mail Access Protocol
161	SNMP Simple Network Management Protocol
443	HTTPS – Secure HTTP

Zu einer fertigen Netzwerkkonfiguration ist noch ein DNS-Dienst hilfreich. Dieser Dienst hilft uns, Namen anstelle von IP-Adressen zu verwenden. Also glücklich die, die einen Namensdienst verwenden können, denn sie müssen sich keine Ziffern merken.

## 13.4

### Reine Einstellungssache

Nach so viel Theorie kommt endlich wieder die Praxis. Wir wollen uns im kommenden Teil diejenigen Dateien ansehen, mit denen man das Netzwerk konfigurieren kann, und jene Tools, mit denen man das Netzwerk testen kann.

In der Datei **/etc/services** findet man eine Liste aller „well known ports“. Hier können wir nachlesen, welche Ports von welchen Server-Diensten mit welchen Protokolltypen verwendet werden.

Wie man Netzwerkkarten in das System integriert, haben wir schon im Kapitel 8 kennen gelernt. Nun soll es an die richtige Konfiguration gehen. Dazu ist das Kommando **ifconfig** (interface configuration) das zentrale Tool. Dieses Programm wird

auch beim Booten verwendet, um die Netzwerkkarten zu initialisieren. Gleich einschränkend dazugesagt: Einstellungen, die mit `ifconfig` vorgenommen werden, sind nur temporär. Wenn man mit temporären Änderungen zufrieden ist, kann man diese Einstellungen in den entsprechenden Konfigurationsdateien ablegen.

Der Befehl `ifconfig`, ohne Argumente aufgerufen, zeigt Informationen aller in dem System eingebauten und aktiven Netzwerkkarten an.

```
#ifconfig
eth0      Link encap:Ethernet  HWaddr 00:10:B5:DA:AA:9D
          inet addr:192.168.1.100  Bcast:192.168.1.255
          Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4902 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2232 errors:0 dropped:0 overruns:0 carier:0
          collisions:0 txqueuelen:100
          RX bytes:2759440 (2.6 MiB)  TX bytes:507767 (495.8
          Interrupt:10 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:345 errors:0 dropped:0 overruns:0 frame:0
          TX packets:345 errors:0 dropped:0 overruns:0 carier:0
          collisions:0 txqueuelen:0
          RX bytes:19569 (19.1 KiB)  TX bytes:19569 (19.1 KiB)
```

Die Ausgabe liefert uns ein Interface ***eth0***, das unserer Netzwerkkarte entspricht. Das zweite Interface, das angezeigt wird, ist ***lo*** (loopback Device) und wird für interne Zwecke verwendet und mit localhost gleichgesetzt. Auf jedem Rechner, der TCP/IP implementiert hat, gibt es das Interface `lo` für localhost. Die `lo`-Interfaces haben immer eine Adresse, die im Netzwerk 127.0.0.0/8 liegt. Es muss aber nicht notwendigerweise 127.0.0.1 sein.

**Tabelle 164: Optionen für ifconfig**

<b>Optionen</b>	<b>Bedeutung</b>
interface	Der Name des zu bearbeiten- den Netzwerkinterface.
up	Das Interface wird aktiviert.
down	Das Interface wird deaktiviert.
-arp	Aktiviert oder deaktiviert den Gebrauch von arp auf dem angegebenen Interface.
-promisc	Aktiviert oder deaktiviert den promiscuous-Modus auf dem Interface.
metric N	Die Metrik des Interfaces wird auf N gesetzt.
mtu N	Der Parameter setzt die MTU (Maximal Trasfer Unit) des In- terfaces.
netmask ADR	Setzt die Netzwerkmaske für das Interface auf ADR.
irq ADR	Setzt den Interrupt auf ADR.
io_addr ADR	Setzt die I/O-Adresse auf ADR.
media TYPE	Setzt den Medientyp auf TYPE.
-broadcast ADR	Setzt die Broadcast-Adresse auf ADR.
-pointopoint ADR	Aktiviert den PPP-Modus der Netzwerkkarte zur direkten Anbindung an einen anderen Rechner.
hw CLASS ADR	Setzt die Hardware-Adresse (MAC-Adresse) des Interfaces.
ADR	Die IP-Adresse des Interfaces.



```
#ifconfig eth0 192.168.0.1 netmask 255.255.255.0 /  
broadcast 192.168.0.255 mtu 1500
```

Damit wird das Interface eth0 mit der Adresse 192.168.0.1/24 und einer maximalen Transfer-Unit gesetzt.

```
#ifconfig eth0 down
```

Damit wird das Interface eth0 abgeschaltet. Mit der Option up kann man es wieder aktivieren.

### Nachtrag

Bei jeder Klasse von IP-Adressen ist ein Bereich definiert, der nicht öffentlich ist und für die Verwendung im eigenen Netzwerk dient. Diese privaten Adressen werden nicht in das Internet geroutet und werden von Internetroutern auch verworfen.

```
10.0.0.0      - 10.255.255.255  
172.16.0.0   - 172.31.255.255  
192.168.0.0  - 192.168.255.255
```

Diese Adressenbereiche können innerhalb einer Firma oder eines Home-Netzwerkes verwendet werden.

Mit dem Befehl **route** kann man sich die Routen, die auf dem System konfiguriert worden sind, ausgeben lassen und verändern. Wir sind im Netzwerk 192.168.1.0/24, und unsere IP-Adresse lautet 192.168.1.100.

```
# route  
Kernel IP routing table  
Destination  Gateway  Genmask   Flags Metric Ref    Use Iface  
localnet      *    255.255.255.0  U        0      0        0 eth0
```

Damit erhalten wir eine Liste von bekannten Routen durch das Netzwerk. Wir haben hier nur eine einzige Route, und das ist unser lokales Netzwerk. Wenn wir auch andere Netze erreichen wollen, brauchen wir auch eine Default-Route.

**Tabelle 165: Optionen für route**

<b>Optionen</b>	<b>Bedeutung</b>
-n	Erzeugt eine numerische Ausgabe der Adressen anstelle von symbolischen.
Del	Löscht eine Route.
Add	Fügt eine neue Route hinzu.
target	Zielhost oder Zielnetzwerk.
-net	Zielnetz.
-host	Zielrechner.
netmask M	Netzwerkmaske M.
gw GW	Gatewayaddress. Adresse, über der das Ziel erreichbar ist.
metric M	Setzt die Metrik in der Routing-Tabelle.
reject	Setzt eine blockierende Route, welche eine Routensuche fehlschlagen lässt.
dev IF	Angabe für das verwendete Interface.

```
#route add default gw 192.168.1.1
#route
Kernel IP routing table
Destination  Gateway    Genmask    Flags Metric Ref    Use
Iface
localnet      *        255.255.255.0  U      0      0      0 eth0
default    edu01srv.popper 0.0.0.0  UG     0      0      0 eth0
```

Die Einstellung eines DNS-Servers, erfolgt über die Datei **/etc/resolv.conf**. Diese Datei konfiguriert den Kerneldienst **resolver**. Dieser ist dafür zuständig, dass der DNS-Server abgefragt wird und dem Namen eine IP-Adresse zugeordnet (resolving) werden kann. In dieser Datei sind folgende Einträge notwendig:

```
DNS-Server 192.168.1.1
search powerup.at
```

Der erste Eintrag definiert die IP-Adresse des Rechners, auf dem der DNS-Dienst läuft. Der zweite Eintrag dient dazu, dass man nicht immer den vollen Namen – **FQDN** (fully qualified Domain Name) – eingeben muss. Ein FQDN hat die Form rechner-name.domain.topleveldomain.

Bei manchen Systemen wird der Resolver auch über die Datei **/etc/host.conf** eingestellt. Die Form dieser Datei ist:

Keyword      Eintrag

**Tabelle 166: Werte in der Datei host.conf**

<b>Keyword</b>	<b>Bedeutung</b>
order	Bestimmt die Reihenfolge, in der die Auflösung durchgeführt werden soll.
trim	Hierbei wird der Hostname von dem gegebenen Domainname getrimmt.
multi	Werte sind on oder off. Bei on werden alle gültigen Adressen für eine Domain angezeigt.
nospoof	On oder off. Bei on wird ein hostname spoofing verhindert und somit die Sicherheit erhöht.

Jetzt kann man seine Netzwerkkonfiguration testen. Am besten und einfachsten kann man die Netzwerkverbindungen mit dem Kommando **ping** überprüfen.

Mit dem Befehl ping wird eine „Echo“-Aufforderung an einen Rechner gesendet und ein „Echo Reply“ erwartet. Das Tool ping arbeitet mit dem **ICMP** (Internet Control Message Protocol)-Protokoll. Es wird dabei gemessen, ob und wie schnell die Antwort vom fremden Rechner wieder zurückkommt.

Nehmen wir an, unser Rechner ist im Netzwerk 172.26.24.0 mit der Netzwerkmaske 255.255.255.0. Als IP-Adresse ist 172.26.24.23 mit dem Standardgateway 172.26.24.4 konfiguriert.

*Frage:*

Um welche Klasse Netzwerk handelt es sich dabei, und was wurde mit ihr gemacht?

Als DNS-Server wollen wir unseren eigenen Rechner mit der IP-Adresse 172.26.24.23 verwenden (zur Konfiguration des DNS-Dienstes kommen wir etwas später).

*Frage:*

Wie würde man diese Konfigurationen vornehmen, und in welchen Dateien würde man Einträge dafür finden können?

Wir wollen testen, ob unser Rechner mit dem Standardgateway (das ist jene Adresse, an die – wie wir bereits wissen – alle Pakete geschickt werden, die nicht im lokalen Netzwerk direkt erreicht werden können) Verbindung aufnehmen kann, also unser Netzwerk bis dahin einmal funktioniert.

```
#ping 172.26.24.4
```

```
PING 172.26.24.4 (172.26.24.4) 56(84) bytes of data.  
64 bytes from 172.26.24.4: icmp_seq=1 ttl=254 time=0.961 ms  
64 bytes from 172.26.24.4: icmp_seq=2 ttl=254 time=0.931 ms  
64 bytes from 172.26.24.4: icmp_seq=3 ttl=254 time=0.945 ms  
64 bytes from 172.26.24.4: icmp_seq=4 ttl=254 time=0.945 ms
```

```
--- 172.26.24.4 ping statistics ---
```

```
4 packets transmitted, 4 received, 0% packet loss, time  
3003ms rtt min/avg/max/mdev = 0.931/0.945/0.961/0.032 ms
```

Die Ausführung des ping-Kommandos läuft, im Gegensatz zu Windows-Systemen, endlos weiter und kann mit der Tastenkombination STRG + C abgebrochen werden.

Wir erhalten in der Ausgabe die IP-Adresse, an die die Anfrage gestellt wird. Ferner erhalten wir die ICMP-Sequenznummer die TTL (Time to Life) so wie die Zeit, die bei uns unter einer Millisekunde liegt. Die Angaben bei einem Linux-System sind wesentlich genauer als bei der entsprechenden Windows-Implementation, die in diesem Fall nur noch kleiner 10 Millisekunden anzeigen würde. Ebenso finden wir in der letzten Zeile die **rrt** (round trip time). Die rrt ist jene Zeit, die ein Paket bis zum Ziel und wieder zurück benötigt.

**Tabelle 167: Optionen für ping**

<b>Optionen</b>	<b>Bedeutung</b>
-b	Erlaubt das „Anpingen“ einer Broadcast-Adresse (bei uns etwa 172.26.24.255).
-c NUM	Sendet nur NUM Anzahl Echo Anfragen und stoppt dann automatisch.
-f	Bedeutet Flood ping. Hier wird anstelle der ausführlichen Ausgabe für jedes Echo_Request ein . gezeichnet und für jedes Echo_Response ein Backspace. Man erkennt dann grafisch die Güte des Kanals.
-i INTERVALL	Hier wird INTERVALL Sekunden zwischen den einzelnen Anfragen gewartet.
-n	Numerische Ausgabe.
-r	Hierbei wird die Routing-Tabelle umgangen und direkt an einen an das Netzwerkinterface angeschlossenen Rechner gesendet.
-s SIZE	Schickt SIZE große Pakete. Default-einstellung ist 56, welche zu 64 Datenbytes umgewandelt werden. (Das Maximum ist 65507 – wurde beim Teardrop oder ping of death überschritten und führte bei früheren Windows-Systemen zur Netzwerkunterbrechung bzw. Absturz).

Bei Fehlschlägen von ping würden wir die Meldungstypen des ICMP-Protokolls angezeigt bekommen.

```
#ping 192.168.0.1
```

```
PING 172.26.24.55 (172.26.24.55) 56(84) bytes of data.
```

```
From 172.26.24.23: icmp_seq=1 Destination Host Unreachable
```

```
From 172.26.24.23 icmp_seq=1 Destination Host Unreachable
```

```
From 172.26.24.23 icmp_seq=2 Destination Host Unreachable
From 172.26.24.23 icmp_seq=3 Destination Host Unreachable
```

```
--- 172.26.24.55 ping statistics ---
```

```
3 packets transmitted, 0 received, +4 errors, 100% packet
loss, time 1999ms
```

Würde der ping-Befehl auf das Defaultgateway fehlschlagen, sollte man zunächst ein ping auf die eigene IP-Adresse senden. Antwortet diese auch nicht, dann auf das Loopback-Device. Antwortet dieses, funktioniert der TCP/IP-Stack, und man kann mit ziehmlicher Sicherheit davon ausgehen, dass man eine Misskonfiguration auf der Netzwerkkarte vorliegen hat. Man verwendet ping eigentlich als Tool beim Troubleshooting.

Ein weiteres wichtiges Tool, das uns beim Troubleshooting behilflich sein kann und wird, ist das Kommando **tracert**. Das Kommando tracert zeigt uns den Weg, den ein Paket bei seinem Weg durch das Netzwerk nimmt. Somit können wir feststellen, bei welchem „Netzwerkknoten“ die Pakete und damit die Kommunikation unterbrochen wird.

```
#tracert -n www.lpi.org
```

```
tracert to www.lpi.org (24.215.7.109), 30 hops max,
40 byte packets
```

```
 1  172.26.24.4  0.981 ms   1.515 ms   1.954 ms
 2  172.19.64.134 20.492 ms  28.541 ms  56.457 ms
 3  * * *
 4  * * *
 5  * * *
 6  195.22.215.41 74.614 ms  79.723 ms  84.809 ms
 7  195.22.216.22 196.120 ms 200.878 ms 205.553 ms
 8  157.130.222.169 187.789 ms 193.357 ms 198.502 ms
 9  152.63.16.86 127.331 ms 131.804 ms 136.219 ms
10  152.63.0.169 142.206 ms 147.026 ms 150.675 ms
11  152.63.2.101 180.146 ms 185.334 ms 189.086 ms
12  152.63.2.78 193.998 ms 198.825 ms 203.062 ms
13  152.63.131.141 208.135 ms 213.021 ms 217.070 ms
14  208.217.112.78 222.542 ms 226.495 ms 231.620 ms
15  216.191.65.241 171.442 ms 176.374 ms 180.526 ms
16  216.191.67.66 174.179 ms 179.043 ms 184.365 ms
```

```

17 216.191.68.30 178.857 ms 183.530 ms 187.696 ms
18 216.185.67.46 179.152 ms 182.517 ms 187.165 ms
19 216.185.64.22 178.905 ms 183.726 ms 187.135 ms
20 24.215.7.110 186.064 ms 190.875 ms 195.274 ms
21 24.215.7.109 186.046 ms 190.124 ms 194.947 ms

```

Die Ausgabe von `traceroute` zeigt uns in diesem Fall den Weg, den unsere Pakete von unserem Rechner bis zum Web-Server des Linux Professional Institutes nehmen bzw. genommen haben. Die Option `-n` fordert das Kommando dazu auf, die Ausgabe numerisch, anstelle der Namensauflösung, durchzuführen. Als weitere Angabe erhalten wir wieder die `rrt`. Damit man ein besseres Abbild der wirklichen Übertragungsleistung bekommt, werden diese Pakete drei mal gesendet. Die erste Zahl ist die Nummer des Routers, an dem unser Paket vorbei gekommen ist. Unser Paket benötigt also 20 Router, um das Ziel (21) zu erreichen. Diese Zahl wird auch als Hops bezeichnet – also 21 Hops bis an Ziel.

**Tabelle 168: Optionen für `traceroute`**

<b>Optionen</b>	<b>Bedeutung</b>
-6 oder -4	Damit kann man festlegen, ob IP-Adressen der Version 4 oder 6 verwendet werden sollen.
-F	Das „Don’t fragment“-Bit wird gesetzt.
-I DEV	Definiert das Interface, durch das <code>traceroute</code> die Pakete senden soll.
-m MAX	Definiert die maximale Anzahl von Hops.
-n	Numerische Ausgabe anstelle aufgelöster Namen.
-w SEC	Wartet SEC zwischen den Anfragen.

Wenn man `traceroute` öfter hintereinander auf ein und dasselbe Ziel anwendet, wird man unter Umständen feststellen, dass sich die Wege nicht gleichen, also unterschiedlich sind. Wir erkennen

dabei eine Designvorgabe vom IP-Protokoll. Denn IP arbeitet verbindungslos (send and pray) und die Router, die die Netzwerke miteinander verbinden, entscheiden selbstständig, welcher Weg zu genau dieser Zeit am optimalsten wäre und schlägt diesen ein.

Da wir schon beim Troubleshooting sind, soll auch gleich der Befehl **netstat** (Netzwerk Status) genannt sein.

Das Kommando netstat gibt uns Informationen über Netzwerkverbindungen (Unix-Streams, TCP/IP und UDP), Routing-Tabellen und Interface-Statistiken. Meine Empfehlung ist: Wenn das System neu aufgesetzt wurde und im „reinen“ Urzustand ist, sollte man sich mit netstat die Verbindungen bzw. die Dienste, die am Netzwerk lauschen, ansehen. Wenn man sich später einmal nicht mehr sicher ist, ob man auf dem System z. B. ein Trojanisches Pferd installiert bekommen hat, kann man einfach mit netstat nachsehen, ob vielleicht ein Dienst auf dem Netzwerk lauscht, den wir damals beim ersten Mal nicht gesehen haben. Dies kann schon ein wichtiger Hinweis darauf sein, dass man einen Trojaner laufen hat.

### #netstat

Active Internet connections (w/o servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	tux.local:netBIOS-ssn	172.26.24.66:iad1	ESTABLISHED
tcp	0	0	tux.local:5901	172.26.24.111:3674	ESTABLISHED
tcp	0	0	tux.local:netBIOS-ssn	172.26.24.111:3681	ESTABLISHED

Active UNIX domain sockets (w/o servers)

Proto	RefCnt	Flags	Type	State	I-Node	Path
unix	12	[ ]	DGRAM		1313	/dev/log
unix	2	[ ]	DGRAM		470656	
unix	3	[ ]	STREAM	CONNECTED	466695	
unix	3	[ ]	STREAM	CONNECTED	466689	/tmp/.X11-
unix/X0						
unix	3	[ ]	STREAM	CONNECTED	466688	
...						

Wir bekommen die Ausgabe über Dienste, die auf das Netzwerk lauschen und welchen Zustand diese Dienste haben. Bei unserer obigen Ausgabe haben wir einen Samba-Server laufen, der ge-



rade aktive Verbindungen zu den Rechnern mit der IP-Adresse 172.26.24.111 und 172.26.24.66 besitzt. Der verwendete Port wird dabei durch einen Doppelpunkt von der IP-Adresse getrennt dargestellt. Die Port-Nummern (und damit die eigentlichen Dienste oder Services) werden aufgelöst und als Namen angezeigt. Mit der Option `-n` kann man dies verhindern.

Mit dem Befehl `netstat` und der Option `-r` erhalten wir die gleiche Ausgabe wie mit dem Befehl `route`. Also die Kernel-Routing-Tabelle.

```
#netstat -r
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
172.26.24.0 * 255.255.255.0 U 0 0 0 eth0
default 172.26.24.4 0.0.0.0 UG 0 0 0 eth0
```

```
#netstat -i
Kernel Interface table
Iface MTU RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR flg
eth0 1500 227654 0 0 0 221759 0 0 0 BMRU
lo 16436 1111 0 0 0 1111 0 0 0 LRU
```

**Tabelle 169: Optionen für netstat**

Optionen	Bedeutung
-r	Zeigt die Routing-Tabelle an.
-i	Zeigt Informationen über alle Netzwerkschnittstellen an.
-s	Gibt eine Statistik aus.
-n	Numerische Ausgabe. Es können auch --numeric-host --numeric-port angegeben werden. Nur der Rechnername oder der Port wird davon beeinflusst, alle anderen Angaben werden namentlich angezeigt.

--protocol=T	Es werden nur Informationen über den Typ T des Protokolls ausgegeben. (Typen können sein: inet, unix, ipx, ...).
-C	Kontinuierliche Ausführung von netstat (Refresh alle Sekunden).
-a	Zeigt alle Informationen. Alle lauschenden und nicht lauschenden Services werden angezeigt.

```
#netstat -na
```

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:5801	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:139	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:5901	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:111	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:80	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:6000	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:6001	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:631	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:953	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:25	0.0.0.0:*	LISTEN
tcp	0	0	172.26.24.23:139	172.26.24.66:1030	ESTABLISHED
tcp	0	0	172.26.24.23:5901	172.26.24.111:3674	ESTABLISHED
tcp	0	0	172.26.24.23:139	172.26.24.111:3681	ESTABLISHED
tcp	0	0	:::53	:::*	LISTEN
tcp	0	0	:::22	:::*	LISTEN
udp	0	0	172.26.24.23:53	0.0.0.0:*	
udp	0	0	127.0.0.1:53	0.0.0.0:*	
udp	0	0	0.0.0.0:32871	0.0.0.0:*	
udp	0	0	127.0.0.1:32874	0.0.0.0:*	
udp	0	0	127.0.0.1:32875	0.0.0.0:*	
udp	0	0	0.0.0.0:111	0.0.0.0:*	
udp	0	0	0.0.0.0:631	0.0.0.0:*	
udp	0	0	:::53	:::*	
udp	0	0	:::32872	:::*	

Active UNIX domain sockets (servers and established)

Proto	RefCnt	Flags	Type	State	I-Node	Path
unix	2	[ ACC ]	STREAM	LISTENING	10507	/tmp/mcopen
...						

Ein letztes Tool, ist das **tcpdump**-Kommando. Mit tcpdump wird das Netzwerkinterface in den sogenannten **Promiscuous**-Modus versetzt, was bedeutet, dass alle Pakete, die auf dem Netzwerk transportiert werden, auch als Kopie in unser Interface eingelese werden. Dies auch, wenn das Paket gar nicht für uns bestimmt ist. Wir können damit den gesamten Netzwerkverkehr mitprotokollieren. Sie werden sicher die Risiken solcher Methoden erkennen. Natürlich unter der Voraussetzung, dass sie nur zum Zwecke der Fehlerfindung eingesetzt werden, ist uns das tcpdump und ähnliche Werkzeuge sehr hilfreich.

```
#tcpdump
```

```
tcpdump: listening on eth0
```

```
10:02:57.396280 tux.local.netBIOS-ssn > 172.26.24.111.4178: P
2232860734:2232860809(75) ack 3795972909 win 33232 NBT Packet (DF)
[tos 0x10]
```

```
10:02:57.396571 172.26.24.111.4178 > tux.local.netBIOS-ssn: P 1:46(45)
ack 75 win 64701 NBT Packet (DF)
```

```
10:02:57.396644 tux.local.netBIOS-ssn > 172.26.24.111.4178: P
75:114(39) ack 46 win 33232 NBT Packet (DF) [tos 0x10]
```

```
10:02:57.397130 tux.local.filenet-nch >
WS01IS01.highway.telekom.at.domain: 13518+ PTR? 111.24.26.172.in-
addr.arpa. (44) (DF)
```

```
10:02:57.397612 172.26.24.111.4178 > tux.local.netBIOS-ssn: P
46:131(85) ack 114 win 64662 NBT Packet (DF)
```

```
10:02:57.397726 tux.local.netBIOS-ssn > 172.26.24.111.4178: P
114:218(104) ack 131 win 33232 NBT Packet (DF) [tos 0x10]
```

```
10:02:57.398006 172.26.24.111.4178 > tux.local.netBIOS-ssn: P
131:225(94) ack 218 win 64558 NBT Packet (DF)
```

```
10:02:57.398052 tux.local.netBIOS-ssn > 172.26.24.111.4178: P
218:325(107) ack 225 win 33232 NBT Packet (DF) [tos 0x10]
```

```
10:02:57.398218 172.26.24.111.4178 > tux.local.netBIOS-ssn: P
225:313(88) ack 325 win 64451 NBT Packet (DF)
```

```
10:02:57.402299 tux.local.5901 > 172.26.24.111.4162: .
2259689511:2259690971(1460) ack 2733572041 win 5840 (DF)
```

```
10:02:57.402315 tux.local.5901 > 172.26.24.111.4162: . 1460:2920(1460)
ack 1 win 5840 (DF)
```

```
10:02:57.402772 172.26.24.111.4162 > tux.local.5901: . ack 2920 win
65535 (DF)
```

```
10:02:57.402794 tux.local.5901 > 172.26.24.111.4162: . 2920:4380(1460)
ack 1 win 5840
```

```
. . .
```

```
. . .
```

```
152 packets received by filter
```

```
56 packets dropped by kernel
```

Die Ausgabe kann nur mit der Tastenkombination Strg + C unterbrochen werden. Standardmäßig „horcht“ der Befehl `tcpdump` auf die erste eingebaute Netzwerkschnittstelle. Wenn mehr als ein Netzwerkinterface eingebaut ist, kann man optional mit angeben, welches Interface benutzt werden soll.

Wir erhalten mit dem `tcpdump`-Befehl eine Ausgabe, die uns die Pakete auflistet, die auf dem Netzwerk, auf dem das angegebene Netzwerkinterface (NIC) angeschlossen ist, transportiert werden. Wir können somit im Fehlerfall auf Netzwerk- und Protokollebene mitprotokollieren, welche Pakete unserem fehlerhaften Dienst entsprechend die Karte verlassen bzw. für unseren Server bestimmt sind.

Zusammengefasst: Wir können mit dem Protokollwissen und dem KnowHow, was der Dienst auf Netz- und Protokollebene in welcher Reihenfolge mit welchen Inhalten eigentlich senden und empfangen soll, überprüfen, ob alles seine Richtigkeit hat bzw. eine entsprechende Fehlerursache finden.

**Tabelle 170: Optionen für `tcpdump`**

<b>Option</b>	<b>Bedeutung</b>
-c NUM	Stoppt nach NUM-Paketen.
-e ALGO	Um IPSEC-Pakete zu entschlüsseln. ALGO gibt den Verschlüsselungsalgorithmus an. Standard ist des-cbc.
-f	Die fremde IP-Adresse wird numerisch anstelle eines Namens ausgegeben.
-i DEV	Es wird auf das Interface DEV gehört.

-n	Konvertiert keinerlei Adressen in Namen.
-t	Es wird die Ausgabe des Zeitstempels pro Zeile verhindert.
-v	Verbose bietet mehr Informationen.
-vv	Noch mehr Informationen.
-vvv	Selbsterklärend ;-)

## 13.5

### Nützliche Netzwerkdienste

Im folgenden Abschnitt wollen wir uns nützliche Netzwerktools, von denen wir einige schon durch den Umgang mit dem Internet kennen, ansehen. Benutzt haben wir ein paar dieser Tools schon öfters, aber heute wollen wir auch hier wieder unserem „Mechanikeransatz“ nachkommen und unter die (grafische) Oberfläche blicken. Starten wir deshalb gleich mit dem **ftp**-Dienst. Wir wollen einen FTP-Server aufsetzen, konfigurieren und testen.

Der FTP-Server ist ein Standarddienst, mit dem es möglich ist, Dateien zwischen Rechnern hin und her zu kopieren. Dies ist natürlich nicht so komfortabel wie die Windows- oder die **NFS**-Freigaben.

Dieser Dienst ist – wie sollte es anders sein – gleich zu Beginn eine Ausnahme. Denn dieser Server benötigt zwei Ports. Alle anderen Dienste finden mit einem Port das Auslangen.

#### Wiederholung

Ein Port ist jene interne Adresse eines Rechners (nicht die IP-Adresse), mit der er auf das Netzwerk hört und Daten annehmen und austauschen kann. Kurz zusammengefasst kann man die IP-Adresse mit der Hausnummer eines Rechners vergleichen und die Port-Nummer mit der Türnummer in dem Haus selbst. Wenn ich in die Küche von Stefan möchte, muss ich in die Hummelstrasse 4 und dort durch die Tür links im Erdgeschoss. Nun bin ich in der Küche. In der Computerei hat dies dann einfach die Form IP-Adresse:Portnummer z. B. 111.222.333.123:7765.

Jeder Dienst (auch Prozess, Service oder schlichtweg Server genannt) benötigt einen Port, um zu kommunizieren. Dieser Port repräsentiert auch den Dienst selbst. Das World Wide Web wird standardmäßig mit dem Port 80 angesprochen. Die Ports und damit die Serverrepräsentationen finden wir in der Datei `/etc/services`. Nun bildet der FTP-Server eine Ausnahme, indem er zwei Ports zur Kommunikation benötigt und zwar den **Port 21** für die Steueranweisungen und den **Port 20** für den eigentlichen Datenaustausch. Nun ist der FTP-Dienst noch ein wenig komplizierter, da er zwei Modi kennt - den **aktiven** und den **passiven**.

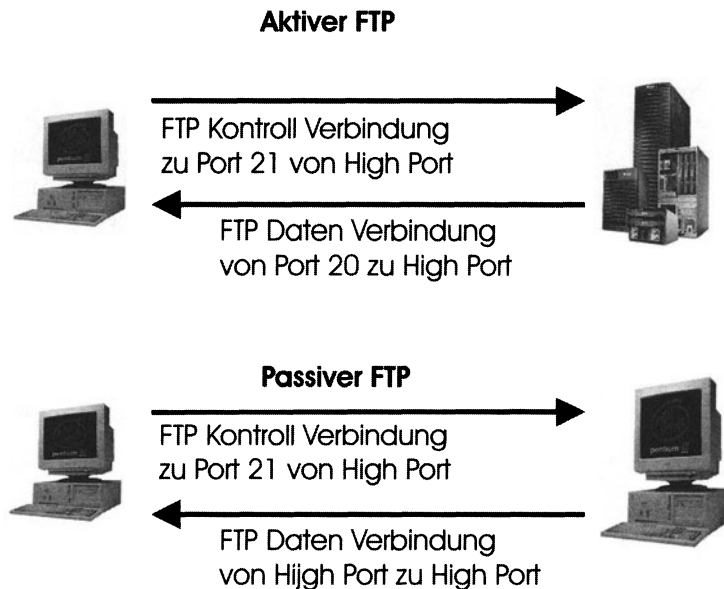


Abbildung 34: Aktives/passives FTP

Was ist der Unterschied zwischen passiv und aktiv?

Beim aktiven Modus nimmt der FTP-Client mit dem Server über Port 21 Kontakt auf und sendet über diesen Kanal die Steuerinformationen wie z. B. `ls`. Jedesmal, wenn der Client Daten vom Server anfordert (wie dies bei `ls` ja der Fall ist), initiiert der Server einen Datentransfer, der immer über den Source-Port 20 zu einem Ziel-Port höher als 1024 (wir erinnern uns an die unprivilegierten Ports) verläuft.

Beim passiven Modus nimmt wieder der Client über Port 21 mit dem Server Kontakt auf. Wenn man wieder mit `ls` Daten vom Server anfordert, wird allerdings der Datentransfer nicht vom Server initiiert, sondern vom Client. Die Port-Nummer des Clients ist immer größer 1024, und nun wird auch die angeforderte Port-Nummer vom Server über 1024 gelegt.

Wir erkennen schon, dass der FTP-Dienst in Verbindung mit einer Firewall problematisch ist. Die derzeit am häufigsten eingesetzte Technik, beim Internet Explorer beispielsweise, ist das passive FTP.

Wir wollen uns im Folgenden die Installation und Konfiguration eines FTP-Servers ansehen. Als Beispiel wird hier der ***vsftpd*** (Very Secure FTP Daemon)-Server verwendet. Alle anderen FTP-Dienste werden analog behandelt. Allerdings sind in der Funktionsvielfalt doch erhebliche Unterschiede zu finden. Deshalb muss ich an dieser Stelle für die jeweiligen Server auf deren Manualseiten für die Konfigurationsdateien und Optionen verweisen. Die grundsätzliche Vorgehensweise werden wir aber jetzt betrachten.

### Aufgabe

Suchen Sie mit [www.rpmfind.net](http://www.rpmfind.net) den FTP-Server `vsftpd` (Very Secure FTP Daemon), der für Ihr installiertes System am besten passt und installieren Sie ihn mit dem `rpm`-Mechanismus. Wenn Sie ihn schon bei der Systeminstallation mitinstalliert haben, entfällt diese Übung natürlich.

Erledigt. Nun haben wir den Server installiert, aber noch nicht konfiguriert. Wir haben die Möglichkeit, den FTP-Server allen beliebigen Benutzern zur Verfügung zu stellen. Diese Art nennt man anonymous FTP-Server. Aus sicherheitstechnischen Gründen kann ich persönlich nur davon abraten, einen anonymous FTP-Server wirklich in einer produktiven Umgebung zu betreiben.

Die zweite Art, wie man einen FTP-Dienst konfigurieren kann, ist ein User-basierender Ansatz. Wir haben hier Benutzerkennung und Passwort, um den FTP-Dienst benutzen zu können. Welche Art in Ihrer Produktivumgebung eingesetzt werden soll, wird vom Ziel des Services abhängen. Die Konfigurationsdatei, mit der man diese Arten einstellen kann, heißt ***/etc/vsftpd.conf***.

Die Konfigurationsdatei ist in Abschnitte unterteilt. Betrachten wir einmal stellvertretend den Abschnitt für den anonymen FTP.

```
# Anonymus FTP user Settings
#
# Allow anonymous FTP?
#
anonymous_enable=YES
```

Damit wird der anonymous FTP-Dienst gestartet.

```
#
# Uncomment this to allow the anonymous FTP user to upload files.
#This only
# has an effect if the above global write enable is activated. Also,
#you will
# obviously need to create a directory writable by the FTP user.
#
#anon_upload_enable=YES
```

Hier kann eingestellt werden, ob ein User mit Namen anonymous auch Dateien auf den FTP-Server aufspielen (uploaden) darf.

```
#
# Default umask for anonymus users is 077. You may wish to change this
# to #022.
# if your users expect that (022 is used by most other ftpd's)
#
#anon_umask=022
```

Einstellen der Standardpermissions beim Anlegen von Dateien.

```
# Uncomment this if you want the anonymous FTP user to be able to cre-
# ate
# new directories.
#
#anon_mkdir_write_enable=YES
```

Wenn der anonyme Benutzer auch Directories erstellen darf, muss man diesen Eintrag auskommentieren.

```
# If you want, you can arrange for uploaded anonymous files to be
# owned by
# a different user. Note! Using "root" for uploaded files is not
# recommended!
#
#chown_uploads=YES
#chown_username=whoever
```



Hier kann man einstellen, ob der anonyme Benutzer am System einem bestimmten User entsprechen soll. Standard ist der User **ftp**. Wir benötigen also, damit unser FTP-Server als anonymous FTP-Server arbeiten kann, einen User, der z. B. ftp heißt und auf unserem System den anonymen Benutzer repräsentieren soll.

### Aufgabe

Wenn auf Ihrem System kein Benutzer-ftp vorhanden ist, legen Sie einen solchen Benutzer bitte an. Das HOME-Verzeichnis ist jenes, das auch über den FTP-Dienst angesprochen werden kann.

Der vsftpd arbeitet mit dem anonymen Benutzer bzw. im anonymous Modus bereits mit einer **chroot**-Umgebung. Mit chroot kann man ein Unterverzeichnis für den Benutzer so erscheinen lassen, als ob es die Wurzel des Dateisystems wäre. Man kann den Benutzer damit in einem Unterverzeichnis einschließen. Die Erstellung einer chroot-Umgebung ist normalerweise nicht so einfach, denn wie Sie vermuten, muss alles, was benötigt wird, der wirklichen Wurzel des Verzeichnissystems im Unterverzeichnis nachgebildet sein. Wir brauchen also das bin-Verzeichnis für die Befehle, das etc-Verzeichnis für alle Passworte, User und Gruppendaten und noch vieles mehr. Sie sehen, dass uns der vsftpd viel Arbeit erspart.

Nun sind wir so weit, wir können unseren Server starten. Dies passiert normalerweise mit dem sogenannten **inetd** (Internet Super Daemon) oder der neueren Variante dem **xinetd**. Der inetd wird über die Konfigurationsdatei **/etc/inetd.conf** eingestellt. Hier haben die Einträge die Form:

Dienst Sockettyp Protokoll Flags User Serverpfad Argumente

In unserem Fall müssten wir in die Datei **/etc/inetd.conf** folgenden Eintrag, also Zeile, einfügen oder, wenn schon vorhanden, anpassen bzw. auskommentieren:

```
ftp stream tcp nowait root /usr/sbin/vsftpd vsftpd
```

Wir können hier angeben, ob der Serverdienst von inetd über einen sogenannten **TCP-Wrapper** gestartet werden soll. Ein TPC-Wrapper lässt das Starten von Diensten nur für diejenigen Rechner zu, die in seinen Konfigurationsdateien als solche spezifiziert wurden. Dieser TPC-Wrapper heißt **tcpd** und besitzt die

Konfigurationsdateien: */etc/hosts.allow* und */etc/hosts.deny*. Der Zugang zu einem Dienst wird erlaubt, wenn ein passender Eintrag in der Datei */etc/hosts.allow* gefunden werden kann. Ist hingegen ein passender Eintrag in der Datei */etc/hosts.deny* zu finden, wird der Zugang verweigert. Wenn keine Konfigurationsdateien gefunden werden können, wird so vorgegangen, als ob sie leer wären. Wenn nichts Spezielles definiert ist, ist per default der Zugang frei.

Das Format der Einträge in den beiden Konfigurationsdateien sieht nun folgendermaßen aus:

daemon : client : Shell-Kommando

Wenn wir unseren FTP-Dienst über den TPC-Wrapper starten wollen, müssen wir in die Datei */etc/hosts.allow* die Zeile

vsftpd: ALL

eintragen.

Mit diesem Eintrag ist es allen Rechnern möglich, den FTP-Dienst aufzurufen.

Damit das auch funktioniert, muss der Eintrag für den FTP-Dienst in der Datei */etc/inetd.conf* folgendermaßen abgeändert werden:

ftp stream tcp nowait root */usr/sbin/tcpd* vsftpd

Aus Sicherheitsgründen ist es empfehlenswert, in der Datei */etc/hosts.deny* die Zeile ALL: ALL zu haben. Denn das bedeutet, dass alles, was wir nicht explizit in der Datei */etc/hosts.allow* erlauben, standardmäßig verboten ist. Wir können mit diesem Ansatz der Sicherheitslogik eigentlich nichts vergessen. Es ist immer der bessere Weg zu sagen: „Alles, was nicht ausdrücklich erlaubt ist, ist verboten“. Es gibt vordefinierte Pattern, die man in diesen Dateien verwenden kann.

**Tabelle 171: Pattern für die hosts.XYZ Dateien**

<b><i>Pattern</i></b>	<b><i>Bedeutung</i></b>
%a	Die Hostadresse.
%c	Clientinformationen in der Form user@host.

%h	Der Clienthostname oder Adresse.
%s	Serverinformationen in der Form daemon@host.
%u	Der Username des Clients.
%%	Ergibt ein %.

Der xinetd, ist aufgeteilt in eine Hauptkonfigurationsdatei und in dienstbezogene Konfigurationsdateien, die in einem eigenen Unterverzeichnis gespeichert sind.

Die Hauptdatei lautet **/etc/xinetd.conf** und sieht folgendermaßen aus:

```
#
# xinetd.conf
#
# Copyright (c) 1998-2001 SuSE GmbH Nuernberg, Germany.
# Copyright (c) 2002 SuSE Linux AG, Nuernberg, Germany.
#

defaults
{
    log_type          = FILE /var/log/xinetd.log
    log_on_success    = HOST EXIT DURATION
    log_on_failure    = HOST ATTEMPT
#    only_from        = localhost
    instances         = 30
    cps               = 50 10

#

}

includedir /etc/xinetd.d
```

Hier werden die globalen Einstellungen getroffen, und mit der letzten Zeile wird konfiguriert, wo die dienstbezogenen Dateien zu finden sind. In unserem Fall ist dies das Verzeichnis **/etc/xinetd.d**. Hier finden wir für alle Dienste die entsprechenden Dateien. Für unseren FTP-Dienst wird schon eine Datei mit dem Namen vsftpd installiert. Der Inhalt der Datei ist folgender:

```

# default: off
# description:
# The vsftpd FTP-Server serves FTP connections. It uses
# normal, unencrypted usernames and passwords for authentication.
# vsftpd is designed to be secure.
service ftp
{
    socket_type          = stream
    protocol             = tcp
    wait                 = no
    user                 = root
    server               = /usr/sbin/vsftpd
#    server_args          =
#    log_on_success       += DURATION USERID
#    log_on_failure       += USERID
#    nice                 = 10
    disable              = yes
}

```

Darin finden wir wieder die selben Einträge wie bei der Datei `inetd.conf`, lediglich etwas anders angeordnet und um diverse Einstellungsmöglichkeiten erweitert. Betrachten wir den untersten Eintrag:

```

        disable              = yes

```

Das heißt, dass der Dienst nicht aktiv ist. Wenn wir diesen Eintrag von `yes` auf `no` ändern, haben wir den Internet Super Daemon so konfiguriert, dass er Anfragen für den FTP-Server akzeptiert und diesen Dienst startet. Wir sehen, dass der `inetd` oder der `xinetd` ein Mutterdienst für verschiedene Dienste ist. Wir müssen keine Ressourcen für den FTP-Dienst verschwenden, wenn kein aktueller Bedarf an diesem Dienst besteht. Der `inetd` oder `xinetd` überwacht die Ports der konfigurierten Dienste und startet diese nach Bedarf.

Unsere Konfigurationsdatei für den FTP-Server sieht nun folgendermaßen aus:

```

# default: off
# description:
# The vsftpd FTP-Server serves FTP connections. It uses

```

```
# normal, unencrypted usernames and passwords for authentication.
# vsftpd is designed to be secure.
service ftp
{
    socket_type          = stream
    protocol             = tcp
    wait                 = no
    user                 = root
    server               = /usr/sbin/vsftpd
#    server_args          =
#    log_on_success       += DURATION USERID
#    log_on_failure       += USERID
#    nice                 = 10
    disable              = no
}
```

Damit die Änderung aktiviert wird, wissen wir, dass wir den `inetd` oder `xinetd` neu starten müssen bzw. mit dem HUP-Signal veranlassen, seine Konfigurationsdatei neu einzulesen. Wir erinnern uns, dass die Daemon-Prozesse die Konfigurationsdateien nur beim Start einlesen und bearbeiten.

Damit ist alles perfekt, wir können den FTP-Client starten und uns mit unserem Server verbinden.

Als FTP-Client wollen wir das konsolenbasierende Programm ***ftp*** verwenden. Dieses Tool ist sowohl bei Linux als auch auf Windows-Systemen vorhanden, also immer griffbereit.

```
#ftp 172.26.24.23
Connected to tux.local.
220 (vsFTPd 1.2.0)
Name (172.26.24.23:helmut):
```

An dieser Stelle geben wir unseren Usernamen und das Passwort ein oder melden uns als anonymer User ohne Passwort an:

```
Name (172.26.24.23:helmut): anonymous
331 Please specify the password.
Password:
230 Login successful.
```

Remote System type is UNIX.  
Using binary mode to transfer files.  
ftp>

Mit dem Befehl **help** bekommen wir eine Liste aller verfügbaren Kommandos, die wir über diesen FTP-Client an den Server senden können. Diese gehen alle über den Port 21.

**Tabelle 172: Befehle für den FTP-Client**

<b><i>Befehl</i></b>	<b><i>Bedeutung</i></b>
get DATEI	Lädt eine Datei vom Server in das angegebene Verzeichnis am Client.
put DATEI	Überspielt eine Datei auf den FTP-Server.
ls	Listet den Inhalt des fernen Directories auf.
lpwd	Zeigt das aktuelle Arbeitsverzeichnis am lokalen Rechner an.
pwd	Zeigt das aktuelle Arbeitsverzeichnis am Server an.
mkdir	Erstellt ein Verzeichnis.
mget	Lädt mehrere Dateien.
mput	Spielt mehrere Dateien auf den Server.
delete	Löscht eine Datei vom Server.
binary	Wechselt in den binären Datenübertragungs-Modus.
ascii	Wechselt in den ASCII Datenübertragungs-Modus.
quit	Beendet den FTP-Client.

Der FTP-Client kann auch mit Optionen aufgerufen werden.

**Tabelle 173: Optionen für den FTP-Client**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
-i	Schaltet den interaktiven Modus aus. Es wird beim Laden mehrerer Dateien nicht nachgefragt.
-p	Aktiviert den passiven Modus.
-u URL FILE	Lädt die Datei FILE auf den Server URL.
-n	Autologin mittels Datei .netrc.

Damit können wir einen FTP-Server installieren, konfigurieren und Dateien herauf oder herunter laden.

Ein weiterer Dienst, der über den Internet Super Daemon gestartet wird, ist der **telnet**-Dienst. Mit dem telnet-Client kann man eine virtuelle Verbindung zu einem Server aufbauen, der einem eine Konsole mit Shell zur Verfügung stellt. Man kann also remote auf einem Rechner arbeiten, so, als wäre man direkt vor Ort am Terminal – ein ideales Instrument für die Fernwartung. Allerdings möchte ich in diesem Zusammenhang sofort auf die Sicherheitsrisiken eingehen, die sowohl den FTP- als auch den telnet-Dienst betreffen. Bei beiden Diensten erfolgt die Kommunikation zwischen Server und Client im Klartext. Das heißt, jeder Benutzername und jedes Passwort wird im Klartext über das Netzwerk übertragen. Ein User mit eingeschaltetem tcpdump-Tool kann somit Ihren Benutzernamen und das Passwort in Erfahrung bringen. Man sollte in einer echten produktiven Umgebung besser auf Tools wie die **ssb** (Secure-Shell) und **scp** (secure copy), welche auch mit RSA-Verschlüsselung zusammenarbeiten können, zurückgreifen.

Wir wollen nun den telnet-Server wieder installieren und starten. Zu Beginn, kontrollieren wir im Verzeichnis /etc/xinetd.d, ob vielleicht schon ein Eintrag vorhanden ist, oder sehen Sie in die Datei /etc/inetd.conf und kommentieren Sie die Zeile mit dem telnet-Dienst einfach aus oder passen Sie sie Ihren Gegebenheiten an.

*Aufgabe*

Gehen Sie wieder auf die Seite von `rpmfind` und suchen Sie den Server ***in.telnetd***, der am besten zu Ihrem System passt, und installieren Sie das Paket mit dem `rpm`-Mechanismus. Wenn ein `telnet`-Server auf Ihrem System bereits installiert ist, ist dieser Schritt nicht notwendig.

Nach erfolgreicher Installation können Sie in dem Verzeichnis `/etc/xinetd.d` eine Datei mit dem Namen `telnet` finden. Wenn Sie sich diese Datei wieder ansehen, werden Sie die Ähnlichkeiten mit dem `FTP`-Server erkennen. Sie müssen, um den `telnet`-Server zu aktivieren, wieder die letzte Zeile von `disable = yes` auf `disable = no` ändern und den `xinetd` neu starten.

*Hinweis*

Das Starten des `xinetd` oder des `inetd` erfolgt über die Start-Skripte, die man im Verzeichnis `/etc/init.d` findet. In unserem Fall also `/etc/init.d/xinetd restart`.

Wenn der `telnet`-Server läuft, können wir den `telnet`-Client starten.

```
#telnet 172.26.24.23
Trying 172.26.24.23 . . .
Connected to 172.26.24.23.
Escape character is '^]'.
Welcome to SuSE Linux 9.0 (i586) - Kernel 2.4.21-99-athlon (5).
tux login:
```

Nun wartet der Dienst auf die Eingabe von Benutzername und Passwort. Das bedeutet, dass nur ein Benutzer mit einem gültigen User-Account auf dem Rechner die Möglichkeit besitzt, sich an diesem anzumelden. Wenn die Anmeldung nicht in einem gewissen Zeitfenster erfolgt, wird der `telnet`-Dienst vom Server her unterbrochen und man wird „rausgeworfen“.

Nach erfolgreicher Anmeldung erhält man einen Prompt genauso, als würde man direkt vor dem entfernten Computer sitzen. Man kann hier alle Befehle wie gewohnt eingeben und verarbeiten lassen, den Rechner neu starten oder herunter fahren lassen. Man kann allerdings keine Dateien herauf oder herunter kopieren.

Bisher haben wir immer mit IP-Adressen gearbeitet, aber durch den `DNS`-Dienst stehen uns auch Namen zur Verfügung. Wir wollen jetzt zwar nicht den `DNS`-Server betrachten, aber einige Tools, die mit Namen umgehen können.



Als erstes sei hier der Befehl **host** angeführt. Der host-Befehl ist ein einfaches DNS-Lookup-Tool. Es wird dazu verwendet, um Namen in IP-Adressen zu konvertieren und vice versa. Als Argument wird der Name bzw. die IP-Adresse des gesuchten Netzknotens, in den meisten Fällen ein Server oder ein Clientrechner, angegeben. Das Tool host kann auch mit IPv6-Adressen richtig umgehen. Als weitere Option kann man den zu verwendenden DNS-Server mit angeben.

```
#host www.linux.de
```

```
www.linux.at has address 143.130.20.3
```

**Tabelle 174: Optione für host**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
-a	Alle Informationen
-l	Veranlasst einen Zonentransfer. Dies ist aber in den meisten Fällen vom DNS-Server her nur gewissen Rechnern gestattet.
-t TYPE	Damit kann man den Abfragetyp angeben (CNAME, NS, SOA, SIG, KEY, etc.).

```
#host -a www.linux.at
```

```
Trying "www.linux.at"
```

```
:: ->HEADER<- opcode: QUERY, status: NOERROR, id: 56915
```

```
:: flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 4
```

```
:: QUESTION SECTION:
```

```
:www.linux.at.      IN ANY
```

```
:: ANSWER SECTION:
```

```
www.linux.at.      429IN A  143.130.20.3
```

```
:: AUTHORITY SECTION:
```

```
linux.at.  10629 IN NS  arachne.luga.at.
```

```
linux.at.  10629 IN NS  nsl.wsr.ac.at.
```

```
linux.at.      10629 IN NS lipo.at0.net.
linux.at.      10629 IN NS vishna.hjp.at.

;; ADDITIONAL SECTION:
arachne.luga.at. 8064  IN A  143.130.20.2
nsl.wsr.ac.at.  8032  IN A  143.130.16.8
lipo.at0.net.    170064 IN A  212.186.13.235
vishna.hjp.at.  8064  IN A  62.99.236.99

Received 221 bytes from 195.3.96.67#53 in 25 ms
```

### Wiederholung

Wenn bei dem `host`-Befehl kein DNS-Server spezifiziert wurde, wird jener verwendet, der in der Datei `/etc/resolv.conf` konfiguriert wurde.

Ein weiteres Tool, um Namensauflösungen zu tätigen, ist das Kommando **dig** (domain information groper). Es soll das **nslookup**-Tool ersetzen. Das `dig`-Kommando ist ein sehr leistungsfähiges Tool, um mit einem DNS-Server zu interagieren. Die meisten Administratoren verwenden es, um Fehler in Netzwerkdiensten aufzustoßern. Viele Fehlerarten, die in einem Netzwerk auftreten können, sind Resultat einer fehlerhaften Namensauflösung. Dies auch, wenn eigentlich gar kein DNS-Server verwendet werden sollte. Wenn man `telnet` mit einer IP-Adresse eingibt, besteht keine Veranlassung, ein `reverse`-Lookup durchzuführen. Manche Implementationen, die sich für ganz sicher halten, führen diese IP-Adresse-Namenskonvertierung allerdings dennoch durch, um festzustellen, ob die Anfrage auch von einer registrierten Adresse kommt.

Die Ausgabe des `dig`-Befehls ähnelt dem des `host`-Befehls:

```
#dig www.linux.de
; <<>> DiG 9.2.2 <<>> www.linux.de
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37420
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDI-
TIONAL: 2

;; QUESTION SECTION:
;www.linux.de.      IN A
```

```
:: ANSWER SECTION:
www.linux.de.      86359 IN A   62.111.65.66

:: AUTHORITY SECTION:
linux.de.      86359 IN NS  ns2.headlight.de.
linux.de.      86359 IN NS  ns1.headlight.de.

:: ADDITIONAL SECTION:
ns2.headlight.de. 82938 IN A   62.111.111.111
ns1.headlight.de. 70880 IN A   62.111.62.111

;; Query time: 22 msec
;; SERVER: 195.3.96.67#53(195.3.96.67)
;; WHEN: Wed Oct 29 08:22:08 2003
;; MSG SIZE rcvd: 132
```

Eine vollständige gültige Befehlseingabe ist gegeben durch:

```
dig @server name typ
```

wobei server für den zu verwendenden DNS-Server steht, name für den zu suchenden Netzknoten und typ wieder den Suchtyp repräsentiert.

In welcher Form bzw. Reihenfolge das System an die Namensauflösung herangeht, wird in der Datei **/etc/nsswitch.conf** geregelt.

```
#
# /etc/nsswitch.conf
#
# Legal entries are:
#
#      compat      Use compatibility setup
#      nisplus     Use NIS+ (NIS version 3)
#      nis         Use NIS (NIS version 2), also called
YP
#      dns         Use DNS (Domain Name Service)
#      files       Use the local files
#      db          Use the /var/db databases

# passwd: files nis
# shadow: files nis
```

```
# group:  files nis
hosts:    files dns
networks: files dns
. . .
. . .
```

Hier kann man konfigurieren, in welcher Reihenfolge welche Art der Namensauflösung benutzt wird. Wir finden z. B. bei `hosts` bzw. `networks` die Reihenfolge ***files*** und ***dns*** angegeben. Das bedeutet, wenn wir einen Namen für einen Rechner eingeben, dass zuerst in den lokalen Dateien nachgeschlagen wird und erst bei einem lokalen Misserfolg der DNS-Server befragt wird. Bei einem Netzwerknamen ist es genauso. Die lokalen Dateien, die zuerst abgefragt werden, heißen folgendermaßen: ***/etc/hosts*** und ***/etc/networks***.

Die Datei `/etc/hosts` kann beispielsweise so aussehen:

```
#
# hosts      This file describes a number of hostname-to-ad-
dress
#            mappings for the TCP/IP subsystem. It is mostly
#            used at boot time, when no name servers are run-
ning.
#            On small systems, this file can be used instead of
a
#            "named" name server.
# Syntax:
#
# IP-Address Full-Qualified-Hostname Short-Hostname
#
127.0.0.1 localhost
172.26.24.23  tux.local  tux

# special IPv6 addresses
::1          localhost ipv6-localhost ipv6-loopback
fe00::0      ipv6-localnet
ff00::0      ipv6-mcastprefix
. . .
```

Eine beispielhafte `/etc/networks` Datei:

```
#
# networks   This file describes a number of netname-to-address
```

```
# mappings for the TCP/IP subsystem. It is mostly
# used at boot time, when no name servers are running.
#
loopback 127.0.0.0
verwaltung 172.26.24.0
lager 192.168.0.0

# End.
```

In dieser Datei kann man den Netzwerken Namen vergeben. Wir können so ein Netzwerk „verwaltung“ und ein Netzwerk „lager“ haben.

In diesen Themenkomplex fällt auch der Befehl **hostname**. Mit ihm kann man, wenn man keine Parameter mit angibt, den Namen bzw. den FQD-Namen eines Computers herausfinden.

```
#hostname
tux.local
```

Wenn man allerdings einen Parameter wie z. B. servus mit angibt, wird der Hostname des Rechners auf servus umgestellt, das wir mit der nochmaligen Eingabe von hostname alleine verifizieren können.

```
#hostname servus
#hostname
servus
```

Allerdings ist zu beachten, dass diese Änderung wiederum nur temporär ist und beim nächsten Bootvorgang überschrieben wird. Beim Booten wird der Hostname üblicherweise aus der Datei **/etc/HOSTNAME** ausgelesen und gesetzt. Wenn man also den Hostnamen dauerhaft ändern möchte, muss man die Datei **/etc/HOSTNAME** anpassen.

Wie steht es dann mit den Änderungen, die wir vorhin mit dem **ifconfig**-Befehl eingestellt haben? Sie vermuten richtig - auch diese sind nur temporär und beim nächsten Bootvorgang verloren. Die Netzwerkeinstellungen werden während des Bootvorganges über die sogenannten Netzwerkskripts gesetzt. Diese Skripts findet man in dem Verzeichnis **/etc/sysconfig/network**. Hier finden wir Dateien, die unsere NICs repräsentieren. Bei un-

serem Rechner haben wir z. B. ein Interface mit dem Namen `eth0` und wir finden eine Datei die **`ifcfg-eth0`** heißt.

```
BOOTPROTO='static'  
BROADCAST='172.26.24.255'  
IPADDR='172.26.24.23'  
MTU=''  
NETMASK='255.255.255.0'  
NETWORK='172.26.24.0'  
REMOTE_IPADDR=''  
STARTMODE='onboot'  
UNIQUE='vuMS.IQxIdIhhuH7'
```

In dieser Datei kann man die Netzwerkeinstellungen dauerhaft verändern. Wenn wir z. B. die IP-Adresse ändern wollen, ändern wir den Teil mit `IPADDR` und `NETWORK`. Beim Bootprotokoll können wir auch auf den DHCP-Dienst umschalten, wenn wir in unserem Netzwerk bereits einen DHCP-Server hätten. Dies geschieht einfach, indem man beim Parameter `BOOTPROTO='dhcp'` schreibt. Alle anderen adressbezogenen Einstellungen fallen natürlich weg. Unsere Datei sieht dann in etwa so aus:

```
BOOTPROTO='dhcp'  
STARTMODE='onboot'
```

Eine weitere wichtige Datei in diesem Verzeichnis ist die Datei mit dem Namen `routes`. Hier wären und sind die sogenannten statischen Routen, also die Wege die ein bestimmtes Paket in ein bestimmtes Netzwerk nehmen muss, eingetragen. Bei uns ist hier nur die sogenannte Default-Route zu finden. Sie ist jene Adresse, an die automatisch alle Pakete gesendet werden, die nicht für ein vom Rechner direkt, über seine Netzwerkinterfaces zu erreichendes Netzwerk bestimmt sind.

```
default 172.26.24.4
```

Unser Computer hat als Default-Route den Rechner oder den Netzwerknotenpunkt `172.26.24.4` konfiguriert.

Ein **DHCP** (dynamic host configuration protocol)-Server ist dazu da, um einem Client beim Starten seine gesamten Netzwerkeinstellungen zu übermitteln. Der Administrator erspart sich somit

einen erheblichen Teil der Konfigurationsarbeit an einem Client, und natürlich ist auch die Wartung um vieles leichter.

Der DHCP-Server versorgt den startenden Rechner mit allen wichtigen Netzwerkparametern wie IP-Adresse, Default-Route, Netzmaske, DNS-Server, Hostname u.v.m.

Sehen wir uns die Konfigurationsdatei des DHCP-Servers an. Diese Konfigurationsdatei findet man im Verzeichnis `/etc` und heißt `dhcpd.conf`. Das `d` am Ende kommt wieder von Daemon.

```
# dhcpd.conf
#
```

In diesem Abschnitt werden der Domainname und die DNS-Server konfiguriert.

```
# option definitions common to all supported networks...
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
```

Als nächstes folgt die Einstellung der Lease-Zeit. Die Lease-Zeit ist jene Zeit, die der Server für die vergebene IP-Adresse reserviert und für diesen speziellen Client als gültig erachtet. Der Client fragt nach der Hälfte der Zeit den Server, ob seine IP-Adresse noch gültig ist oder nicht.

```
default-lease-time 600;
max-lease-time 7200;
```

Nun erfolgt die Angabe des Netzwerkes, für das der DHCP-Server seine Konfigurationsdaten versendet. Hier wird auch festgelegt, welcher Adressbereich (**range**) an Adressen vergeben werden soll. Man will mitunter nicht alle Adressen dynamisch vergeben. Als Optionen kann man hier einstellen, welche Router für dieses Netzwerk zu verwenden sind.

```
subnet 10.152.187.0 netmask 255.255.255.0 {
}
```

```
# This is a very basic subnet declaration.
```

```
subnet 10.254.239.0 netmask 255.255.255.224 {
    range 10.254.239.10 10.254.239.20;
```

```
option routers rtr-239-0-1.example.org, rtr-239-0-
2.example.org;
}
```

Man kann die oben eingestellten und damit als global geltenden Einstellungen im **Subnet Bereich** "überschreiben". Die globalen Einstellungen gelten nur dort, wo nichts anderes spezifiziert wurde. So könnte eine alternative Subnet Definition folgendermaßen aussehen:

```
subnet 10.5.5.0 netmask 255.255.255.224 {
    range 10.5.5.26 10.5.5.30;
    option domain-name-servers ns1.internal.example.org;
    option domain-name "internal.example.org";
    option routers 10.5.5.1;
    option broadcast-address 10.5.5.31;
    default-lease-time 600;
    max-lease-time 7200;
}
```

Es ist auch möglich, einen Server mit dhcp zu konfigurieren. Dies klingt im ersten Moment widersprüchlich, denn eigentlich sollte ja ein Server immer die gleiche IP-Adresse erhalten. Bei der dynamischen Konfiguration ist dies nicht möglich, oder?

Doch! Normalerweise werden die IP-Adressen in der Reihenfolge vergeben, wie sich die Rechner über das Netzwerk bei dem Server melden. Mit der folgenden Einstellung wird es uns aber möglich, aufgrund der **MAC** (Media Access Control)-Adresse einem Rechner immer die gleiche IP-Adresse zuzuweisen.

```
host fantasia {
    hardware ethernet 08:00:07:26:c0:a5;
    fixed-address fantasia.fugue.com;
}
```

Gestartet wird der DHCP-Server wieder über die Start-Skripts und nicht über den Internet Super Daemon. Also kurz: mit /etc/init.d/dhcpd start

Voilà - unser DHCP-Server läuft, und da wir ihn für unser Netzwerk konfiguriert haben, können wir die Netzwerkeinstellungen unserer Clients ab sofort zentral verwalten.

### Frage

Wie kann ich die MAC-Adresse eines Linux-Systems und wie die eines Windows-Systems ausfindig machen?



Ich möchte diesen Abschnitt mit dem wichtigen Tool **whois** beschließen. Mit dem Befehl `whois` kann man die whois-Verzeichnisse abfragen und damit Informationen über registrierte Domainnamen erhalten. Wenn wir uns z. B. interessieren, wem die Domain `www.linux.de` gehört, bzw. wie die Kontaktinformationen zu dieser Domain lauten, können wir die Registrierungsdatenbank mit `whois` abfragen.

```
#whois linux.de
% Copyright (c)2002 by DENIC
%
% Restricted rights.
%
%

domain:      linux.de
descr:       Christian xxxxxxxxxxxxxx
descr:       xxxxxxxxxxxxxx Str. 14
descr:       D-xxxxx xxxxxxxxxxxxxx
nserver:     ns1.headlight.de
nserver:     ns2.headlight.de
status:      connect
changed:     20030514 194438
source:      DENIC

[admin-c]
Type:        PERSON
Name:        Christian xxxxxxxxxxxxxx
Address:     xxxxxxxxxxxxxx Str. 14
City:        xxxxxxxxxxxxxx
Pcode:       xxxxx
Country:     DE
Remarks:    ID #205
Changed:     20030506 140641
Source:      DENIC
[tech-c][zone-c]
Type:        ROLE
Name:        xxxxxxxxxxxxxxxxxxxx Hostmaster
Address:     xxxxxxxxxxxxxxxxxxxx 35
City:        xxxxxxxxxxxxxx
Pcode:       xxxxx
Country:     DE
Phone:
Fax:
Email:       hostadm@ssssssssss.de
```

Remarks: ID #9  
Changed: 20030903 233041  
Source: DENIC

Ich habe aus Datenschutzgründen, obwohl im Internet frei zugänglich, die personenbezogenen Daten mit xxxxx unkenntlich gemacht. Wir sehen schon, dass für einen Angreifer der whois-Dienst sehr viele vertrauliche Daten ausgibt. Wir erfahren hier, über welche DNS-Server die Domain verwaltet wird. Wer der administrative und wer der technische Kontakt ist. Dies geht sogar so weit, dass wir Adresse und Telefonnummer der entsprechenden Personen und Organisationen erhalten. Ich kann Ihnen hier nur raten, einen Weg zu finden, der so wenig wie möglich und so viel wie notwendig von Ihnen und Ihrer Organisation verrät.

Wenn man z. B. ein bestimmtes whois-Verzeichnis abfragen möchte, kann man dies mit der optionalen Angabe -h NAME erreichen. Wir könnten z. B. unsere obige Anfrage direkt zum Rechner whois.ripe.net schicken. Unser Aufruf würde dann so aussehen.

```
#whois -h whois.ripe.net
```

In diesem Kapitel möchte ich die wichtigsten Serverdienste einzeln hervorheben und beschreiben. Den einen oder anderen Dienst haben wir in früheren Kapiteln schon verwendet – so wie den DNS-Server. Nach diesem Kapitel werden Sie in der Lage sein, grundlegende Konfigurationen an einem Mail-, Web- und DNS-Server selbst vornehmen zu können sowie weitere wichtige Tools wie die Secure-Shell für die Verwaltung und Administration des Servers anzuwenden.

### 14.1

#### Der DNS-Server

Der DNS-Server ist wohl einer der wichtigsten und zentralsten Server-Dienste, der es uns erst ermöglicht, mit dem Netzwerk bzw. dem Internet umzugehen. Die Hauptaufgabe des DNS-Servers ist, die Namen in IP-Adressen zu konvertieren und umgekehrt. Stellen Sie sich einen Moment vor, Sie müssten sich alle Web-Adressen als 12-stellige Zahl merken. Da sind dann nur noch ein paar Lieblingsseiten drin. Was ist mit all den anderen, die man zwar unregelmäßig, aber doch immer wieder ansurft? Von den E-Mail-Adressen unserer Geschäftsfreunde ganz abgesehen. Schnell, vergessen wir diese grauenhafte Vorstellung wieder! Sehen wir uns den Server an, der uns das Leben um vieles erleichtert.

Der DNS-Server erleidet oft das Schicksal, mit seinem System bezeichnet zu werden. DNS ist die Bezeichnung für **Domain Name System**, der eigentliche Serverdienst hat den Namen **BIND** (Berkeley Internet Name Domain). Das Softwarepaket BIND wird derzeit hauptsächlich in der Version 8 verwendet. Die Vorgängerversion von BIND 8 war BIND 4. Seit Aufkommen des Windows2000-Servers mit seinem Active Directory, wo sich die Clients beim Starten selbstständig in den DNS-Server eintragen, wird immer öfters die Version BIND 9, der diese dynamische Funktionalität bietet, eingesetzt.

Bevor wir uns exemplarisch der Konfiguration des BIND 8 zuwenden, betrachten wir kurz den grundsätzlichen Aufbau des Domain Name-Systems.

Das Domain Name-System implementiert einen hierarchischen Namensraum, der, wie wir es vom Linux Verzeichnisbaum her kennen, eine Wurzel besitzt. Auch der Namensraum ist eine auf den Kopf gestellte Baumstruktur, wobei die Wurzel mit mehreren auf der ganzen Welt verteilten ROOT-Servern repräsentiert wird. Die Blätter und Äste entstehen durch die untergeordneten Namen (Domains und Subdomains).

Betrachten wir zum Beispiel den folgenden Domainnamen **www.math.ethz.ch**. Wir starten ganz oben und bezeichnen die Ebene nach der Wurzel ( . ) mit Top Level Domain. In unserem Fall ist die Top Level Domain bestimmt mit **ch** für die Schweiz. Die nächst tiefere Ebene in der Hierarchie – allerdings dem Ast folgend, auf dem wir uns schon befinden – führt uns zur sogenannten Second Level Domain. In unserem Fall ist dies der Name **ethz** für die Eidgenössische Hochschule in Zürich. Diese Domain ist noch einmal unterteilt und bringt uns zu einer Sub-Domain oder Third Level Domain. In unserem Fall ist dies die Mathematikfakultät, welche durch **math** repräsentiert wird. Was uns nun vorne übrig bleibt, also das **www**, ist nicht die Bezeichnung eines Dienstes, sondern einfach der Rechnername. Es hat sich allerdings eingebürgert, dass Web-Server im Regelfall den Namen **www** und FTP-Server **ftp** heißen.

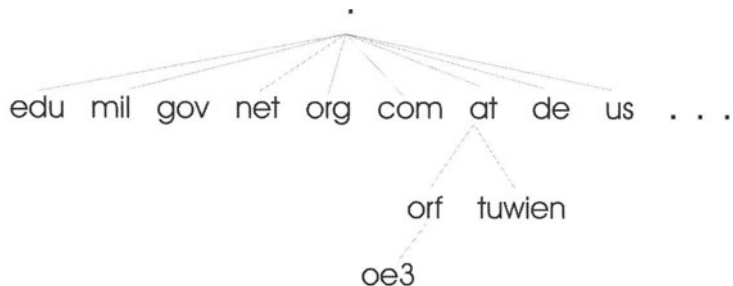


Abbildung 35: DNS-Struktur für **oe3.orf.at**

Der DNS-Server besitzt also Einträge für die Rechner oder Netzwerkknoten der ihm untergeordneten Domäne. Nur für diese eine Domäne, oder auch Zone genannt, ist der DNS-Server zuständig.

Sie runzeln jetzt zu recht die Stirn und fragen sich, wie findet der DNS-Server dann Namen und Adressen von Knoten, für die er nicht zuständig (authoritativ) ist? Ganz einfach. Jetzt kommt die Wurzel der Hierarchie, die ROOT-Server, ins Spiel. Wenn mein benutzter DNS-Server die Antwort auf meine Anfrage nicht kennt, schickt er diese Anfrage an einen der derzeit 16 ROOT-Server weiter. Nun beginnt die Auflösung rekursiv, wie wir es gerade oben exemplarisch durchgeführt haben, zu laufen. Der ROOT-Server kennt alle, die für die TOP Level Domain zuständig sind. Er sendet den Auflösungsauftrag an diese DNS-Server weiter. Diese DNS-Server kennen alle Server, die für alle Second Level Domains für die jeweilige Top Level Domain zuständig sind. Es wird die Anfrage an diese weitergeleitet. Dieser Prozess setzt sich so weiter fort, bis die Anfrage bei einem DNS-Server angelangt ist, der die gewünschte Auskunft geben kann. Damit dieser Prozess nicht ständig ablaufen muss, merkt sich jeder DNS-Server in seinem Cache die Anfragen und die Antworten, sodass bei nochmaliger Abfrage gleich „mein“ DNS-Server die Antwort geben kann.

Der DNS-Server ist auch der einzige Server, der weiß, welcher Rechner für den Empfang und das Versenden von E-Mails einer bestimmten Domain zuständig ist.

Genug der Theorie, wir wollen uns wieder mit vollem Elan an die praktische Konfiguration eines DNS-Servers machen, z. B. für die Domain ***powerup.at***.

Zuerst müssen wir uns versichern, dass das entsprechende Softwarepaket auf unserem Rechner installiert ist.

### Übung

Stellen Sie fest, ob und welche Version von BIND auf Ihrem System installiert ist. Sollte es die Version 9 sein, verzweifeln Sie bitte nicht, sondern lassen es bei dieser Version, da die Konfiguration, die ich Ihnen hier darstellen möchte, für die Version 8 und 9 identisch ist. Wenn keine BIND-Version installiert ist, suchen Sie bitte wieder auf rpmfind nach einer für Ihr System geeigneten Version und installieren Sie sie mit dem rpm-Mechanismus.

Die Hauptkonfigurationsdatei des BIND-8-Systems heißt ***/etc/named.conf***. Die entsprechende Datei für das BIND-4-System heißt ***/etc/named.boot***. Der schnellste Weg, einen Einblick in die Konfigurationsdatei zu bekommen, ist, die wichtigsten Teile daraus zu betrachten und zu kommentieren und an unsere Bedürfnisse anzupassen.

```
# /etc/named.conf
#
# This is a sample configuration file for the name server.  It
```

Im ersten Teil können wir Optionen z. B. für Zugriffsrechte, Weiterleitungen und so weiter definieren. Wir wollen hier nur das Verzeichnis angeben, in dem alle Daten über die Domäne (Zone), die wir mit unserem DNS-Server verwalten wollen, gespeichert werden. Beachten Sie, dass dieses Verzeichnis existieren muss – es wird nicht angelegt. Wir werden unsere Dateien unterhalb des Verzeichnisses `/var/lib/named` speichern. Alle zukünftigen Pfadangaben beziehen sich fortan relativ auf dieses Verzeichnis.

```
options {

    # The directory statement defines the name server's working
    directory

    directory "/var/lib/named";

}

#The first one
# is the definition of the root name servers.  The second one
defines
# localhost while the third defines the reverse lookup for lo-
calhost.
```

Die Domainen, die verwaltet werden, werden auch als Zonen bezeichnet. Die erste Zone, die definiert ist, entspricht der Wurzel des hierarchischen Systems. Mit der Zeile `file „root.hint“` wird die Datei definiert, in der die Namen und die Adressen der ROOT-Server stehen. Diese Zone wird mit `.` bezeichnet.

```
zone "." in {
    type hint;
    file "root.hint";
};
```

Die nächste Zone betrifft das sogenannte LoopBack-Device und damit auch ein Netzwerk mit IP-Adressen. Wir könnten dies löschen, müssen aber nicht. Was wir allerdings bereits bei dieser Definition der Zone erkennen, ist, dass sie in zwei Teile unterteilt ist. Eine sogenannte Forward-Zone und eine Reverse-Zone,

die jeweils wieder in einer eigenen Datei näher spezifiziert sind. Die Beschreibung der Forward-Zone beinhaltet die Auflösung der Namen in IP-Adressen und die Reverse-Zone die Auflösung einer IP-Adresse zu einem Namen.

```
zone "localhost" in {  
    type master;  
    file "localhost.zone";  
};  
  
zone "0.0.127.in-addr.arpa" in {  
    type master;  
    file "127.0.0.zone";  
};
```

Die Zonen-Definition für die Reverse-Zone hat eine Besonderheit. Die Zone **muss** immer die IP-Adresse des Netzwerks ohne den Hostteil in umgekehrter Reihenfolge der Ziffern aufführen, gefolgt von **.in-addr-arpa**.

Der Typen Eintrag gibt an, ob es sich um einen **primary**- oder **secondary**-Server handelt. Der primary-Server wird durch das Schlüsselwort **master** definiert und ist das eigentliche „Arbeitstier“. Der secondary-Server wird durch das Schlüsselwort **slave** definiert und dient nur der Ausfallsicherheit. Wenn der primary-Server einmal ausfallen sollte, kann der secondary-Server die Anfragen übernehmen.

Der file-Eintrag gibt, relativ zum oben konfigurierten Verzeichnis, an, wo die Dateien für die Beschreibung der Zonen liegen.

Wir wollen diese Datei sofort um die Einträge erweitern, die unsere zu verwaltende Zone powerup.at, repräsentieren. Wir definieren die Datei, für die Forward-Zone der Domäne powerup.at mit powerup.at.zone.

```
zone "powerup.at" in {  
    type master;  
    file "powerup.at.zone";  
};
```

Die Reverse-Zone wird durch den Eintrag **24.26.172.in-addr.arpa** definiert und durch die Datei powerup.at.rev beschrieben. Sie sehen, die Dateinamen müssen nicht konform mit den Zonen-Namen sein.

```
zone "24.26.172.in-addr.arpa" in {
    type master;
    file "powerup.at.rev";
};
```

Wir können diese Datei speichern und gehen in das Verzeichnis /var/lib/named, so wie wir es gerade definiert haben. Hier legen wir die zwei Dateien, die wir in der /etc/named.conf Datei für powerup.at angegeben haben, mit folgendem Inhalt an.  
/etc/lib/named/powerup.at.zone:

```
$TTL 1W

@      IN SOA  tux.powerup.at  root.tux.powerup.at (
        42     ; serial (d. adams)
        2D     ; refresh
        4H     ; retry
        6W     ; expiry
        1W )   ; minimum

        IN NS   tux.powerup.at

tux     IN     A  172.26.24.23
www     IN     CNAME tux
```

Die Einträge in dieser Datei bedeuten Folgendes:

Die TTL ist die Time To Live, die mit einer Woche (1W) eingestellt wird. Das heißt, dass ein Eintrag eine Woche lang im Cache gespeichert und dann erst erneuert wird.

Der Eintrag SOA bedeutet Start of Authority, was so viel bedeutet, dass hier der Beginn der Zuständigkeit des Servers für die danach stehende Zone ist (tux.powerup.at – wobei tux der Rechnername ist). Der folgende Eintrag ist die E-Mail-Adresse. Hier ist eine Besonderheit: das @ ist ein Kürzel für ORIGIN und kann deshalb in einer E-Mail-Adresse nicht verwendet werden. Deshalb wird das @ einfach durch einen . ersetzt. Die Mails bekommt also der User root auf dem Rechner tux.powerup.at.

Die Seriennummer ist dazu da, dem secondary-Server mitzuteilen, dass sich die Einträge geändert haben und er die Zone neu aufbauen soll, was mit einem sogenannten Zonentransfer auch



erledigt wird. Dies ist der Mechanismus, wie sich primary- und secondary-Server abgleichen. Mit dem refresh-Eintrag wird bestimmt, wie oft der secondary-Server nachfragt, ob sich die Seriennummer verändert hat oder nicht. Bei uns alle 2 Tage. Wenn der primary-Server gerade nicht verfügbar ist, dann wartet der secondary-Server die Zeitspanne für retry, bis er ein weiteres Mal versucht, die Zone zu transferieren.

Der expiry-Eintrag gibt an, wie lange der secondary-Server mit den Daten des primary-Servers die Arbeit aufrecht erhält – bei uns 6 Wochen. Die Minimum-Angabe definiert, wie lange negative Antworten gecached werden sollen.

Nun kommt die Definition des DNS-Server, also der Eintrag für sich selbst, denn hier haben wir das „Henne Ei“-Problem. Mit dem Eintrag

```
IN NS    tux.powerup.at
```

wird definiert, dass der Rechner mit dem Namen tux.powerup.at der DNS-Server ist. Das NS bedeutet hier Name-Server. Das IN bedeutet, dass es sich um einen Internet Adressen Typ handelt. In weiterer Folge werden die einzelnen Rechner mit Namen versehen und ihnen ihre IP-Adresse zugeordnet.

```
tux  IN    A  172.26.24.23
```

```
www  IN    CNAME tux
```

Das A bedeutet, dass es sich um eine Adressangabe handelt und CNAME ist ein Alias – der sogenannte Canonical Name.

**Tabelle 175: DNS-Record-Typen**

<i><b>Type</b></i>	<i><b>Bedeutung</b></i>
NS	Name-Server.
A	Address – zeigt auf einen Namen.
PTR	Pointer – zeigt auf eine IP-Adresse.
MX	Mail Exchange – zuständiger Rechner für den Mailtransport.

CNAME	Canonical Name – Alias auf einen bereits definierten Rechner.
LOC	Location – definiert die geografische Lage.
TXT	Text – Kommentare.

Betrachten wir die Konfigurationsdatei für die Reverse-Zone.

```
/var/lib/named/powerup.at.rev:
```

```
$TTL 1W
@      IN SOA      tux.powerup.at.  root.tux.powerup.at.
(
    42      ; serial (d. adams)
    2D      ; refresh
    4H      ; retry
    6W      ; expiry
    1W )    ; minimum

    IN NS     tux.powerup.at.

23     IN PTR  tux.powerup.at.
```

Die Datei sieht der ersten zum Verwechseln ähnlich. Die Unterschiede sind klein aber fein. Zunächst merken wir, dass bei `tux.powerup.at` ein abschließender Punkt steht und die nachfolgende E-Mail-Adresse ebenfalls mit einem Punkt schließt. Diese Punkte ziehen sich über alle Domain- oder Rechnerangaben hindurch. Auch der Eintrag für den DNS-Server (NS) schließt mit einem Punkt ab.

Der prinzipielle Unterschied liegt in der Umkehrung der ersten Datei, also die Zuordnung von IP-Adressen zu den Rechner.Domain-Namen. Wir sehen, dass aufgrund der Schreibweise der Zonendefinition nur der Hostteil als IP-Adressangabe genügt, da das Netzwerk ohnehin durch ORIGIN definiert wurde. Bei der Zuordnung IP-Adresse zu Namen ändert sich der Typ von A auf PTR.

Damit haben wir unseren DNS-Server für die Verwaltung der Domäne powerup.at konfiguriert. Wir können nun den Server, der den Namen **named** hat, starten. Dies geschieht wieder über die Start-Skripte in /etc/init.d:

```
#/etc/init.d/named start
```

Damit läuft unser DNS-Server. Um zu kontrollieren, ob der Dienst auch ohne Fehler gestartet wurde, kontrollieren wir die Logdatei /var/log/messages. Hier werden alle Fehler protokolliert, bis hin zu den Zeilennummern in den entsprechenden Konfigurationsdateien. Sieht hier alles in Ordnung aus, wenn also in der Logdatei keine Fehler protokolliert wurden, heißt dies allerdings noch lange nicht, dass unser DNS-Server auch wirklich so funktioniert, wie wir es wollen. Um die richtige Funktionsweise zu überprüfen, verwenden wir unsere Tools von vorhin – dig, host und nslookup.

#### Aufgabe

Passen Sie bitte die Datei /etc/resolv.conf so an, dass Ihr DNS-Server verwendet wird, und testen Sie mit den Tools dig und host die Funktionstüchtigkeit Ihres Services.

#### Hinweis

Würden wir die DNS-Server-Konfigurationsdateien unverändert lassen und das Service starten, hätten wir einen sogenannten caching only Server. Dieser Server dient nur der Abfrage und dem Cachen der Antworten.

## 14.2

### Mail-Server – Kommunikation ist alles

Nachdem wir einen funktionierenden DNS-Server unser Eigenen nennen und damit die Herren einer eigenen Domain sind, wollen wir natürlich auch unsere eigenen Mails verwalten können. Dazu benötigen wir aber unseren eigenen Mail-Server. Der Mail-Server, den wir verwenden wollen, ist der wohl am meisten verbreitete Server - **sendmail**. Bevor wir diesen aber konfigurieren, betrachten wir noch ein paar theoretische Fakten.

Der Mail-Server hört auf den **Port 25** des jeweiligen Rechners für den Empfang und das Versenden der E-Mails. Dieser Teil des Dienstes wird **MTA** (Message Transfer Agent – z. B. sendmail) genannt. Eine eingehende E-Mail wird nun dem **MDA** (Mail Delivery Agent – z. B. procmail) vom MTA übergeben und dem eigentlichen User (gültiger Account auf dem Rechner) zugestellt.

Das Zustellungsverzeichnis liegt üblicherweise im Verzeichnis */var/spool/mail/*, und dort wird eine Datei mit dem Namen des Users erzeugt, und jede neue E-Mail wird an diese Datei angehängt. Letztendlich wird die Mail durch den **MUA** (Mail User Agent – z. B. Outlook) vom Server wieder abgeholt. Wir sehen, dass wir zu einem funktionierenden Mail-System 3 Komponenten benötigen. Die zentrale Hauptkonfigurationsdatei heißt */etc/sendmail.cf*. Eine Liste von Rechnern, denen es gestattet ist das Mail-System zu verwenden, findet man in der Datei */etc/sendmail.cw*. Es gibt noch die Datei */etc/aliases*, in der alle Aliases oder Spitznamen zu den eigentlichen Mailaccounts eingetragen werden. Zu den Aliases und deren Bedeutung kommen wir im Anschluss. Das Softwarepaket sendmail ist wohl jenes, das die umfangreichsten Anpassungsmöglichkeiten an die eigenen Bedürfnisse bietet. Alleine über dieses Programm könnte man Bücher schreiben, die ca. 1000 Seiten umfassen – und diese sind auch schon geschrieben worden. Dass das Sendmail für größtmöglichen Funktionsumfang und leichte maschinelle Verarbeitung anstelle leichter Konfigurierbarkeit entwickelt wurde, erkennt man sofort, wenn man einen Blick in die Konfigurationsdatei sendmail.cf wirft.

```

. . .
SParse0
R<@>      $@ <@>      special case error msgs
R$* : $* ; <@>      $$error $@ 5.1.3 $: "553 List::; syntax illegal
for recipient addresses"
R@ <@ $* >      < @ $1 >      catch "@@host" bogosity
R<@ $+>      $$error $@ 5.1.3 $: "553 User address required"
R$+ <@>      $$error $@ 5.1.3 $: "553 Hostname required"
R$*      $: <> $1
R<> $* < @ [ $* ] : $+ > $* $1 < @ [ $2 ] : $3 > $4
R<> $* < @ [ $* ] , $+ > $* $1 < @ [ $2 ] , $3 > $4
R<> $* < @ [ $* ] $+ > $* $$error $@ 5.1.2 $: "553 Invalid address"
R<> $* < @ [ $+ ] > $* $1 < @ [ $2 ] > $3
R<> $* < $* : $* > $* $$error $@ 5.1.3 $: "553 Colon illegal
in host name part"
R<> $*      $1
R$* < @ . $* > $* $$error $@ 5.1.2 $: "553 Invalid host name"
R$* < @ $* .. $* > $* $$error $@ 5.1.2 $: "553 Invalid host
name"
R$* < @ $* @ > $* $$error $@ 5.1.2 $: "553 Invalid route address"

```

```
R$* @ $* < @ $* > $* $#error $@ 5.1.3 $: "553 Invalid route  
address"  
R$* , $-0 $*   $#error $@ 5.1.3 $: "553 Invalid route address"  
.  
.  
.
```

Und so geht es ca 2000 Zeilen lang dahin ☹.

Bitte lassen Sie jetzt keine Panik aufkommen. Die allermeisten Einstellungsmöglichkeiten benötigt man im Normalfall – Internet Service Provider (ISPs) einmal ausgenommen – ohnehin nicht.

Die wichtigste Funktion für die Prüfung ist das **Smart relay**. Dies wird benötigt, wenn man nicht selbst für die Mails zuständig sein möchte, sondern der Mail-Server des Providers dies übernehmen soll.

Suchen wir den Eintrag für den smart relay host. Bei meinem mir derzeit vorliegenden Konfigurationsfile ist er in der Zeile 103 zu finden:

```
# "Smart" relay host (may be null)
```

```
DS
```

Hier müssen wir den gewünschten Internet Service Provider angeben. Also den Mail-Server unseres ISPs, der die Mails entgegen nimmt.

```
# "Smart relay host (may be null)
```

```
DSemail.aon.at
```

Damit können wir Mails an den Mail-Server unseres ISPs weiterleiten.

Damit die Konfiguration des sendmail.cf-Files leichter wird, wurden sogenannte m4-Macros entwickelt, die aus einem Masterfile (Macro-Datei) eine sendmail.cf-Datei erzeugen können. Dieses Masterfile ist für die Konfiguration einfacher handhabbar.

Jedes der m4-Macros hat die Form:

```
Name(arg1, arg2, arg3, ....)
```

Es gibt eine Unzahl von verschiedenen vordefinierten Makros, die den Umfang des Buches sprengen würden. Ich werde mich deshalb auf ein paar ausgewählte beschränken und verweise Sie gleichzeitig auf die einschlägigen Hilfetexte zum sendmail-Softwarepaket.

Hier ein Auszug aus einer m4-Datei mit dem Namen linux.mc der SuSE 9.0-Distribution:

```

divert(-1)
# Copyright (c) 1997-1999,2000 SuSE GmbH Nuernberg, Germany.
# Author: Florian La Roche
#         Werner Fink         <feedback@suse.de>
#
# After the `divert(0)' all lines starting with `dn1' are
# comments until the next newline character.
# Putting words into ``'-pairs disables macro expansion
#
include(`/usr/share/sendmail/m4/cf.m4')
divert(0)dn1
VERSIONID(`@(#)Setup for SuSE Linux 8.12.3-0.4 (SuSE Linux)
2002/01/14')
dn1
dn1 This is the default configuration for SuSE Linux.
dn1 See `/usr/share/sendmail/ostype/suse-linux.m4' and take a
look
dn1 into `/usr/share/sendmail/README' for more information.
dn1
dn1 The suse-linux.m4 enables the FEATURES mailertable, gener-
icstable,
dn1 virtusertable, and access_db. Just look to those file for
some
dn1 examples. They are stored in `/etc/mail/'. If you have
changed
dn1 one or more files you should run SuSEconfig or generate
the
dn1         `.db'         files         by         hand         (see
/sbin/conf.d/SuSEconfig.sendmail).
dn1
dn1 NOTE: YOU HAVE TO CHANGE THE CONFIGURATION TO FIT YOUR
NEEDS
dn1         BEFORE ACTIVTING SOME OF THESE EXAMPLES!
dn1
OSTYPE(`suse-linux')
FEATURE(`promiscuous_relay')
FEATURE(`always_add_domain')
MASQUERADE_AS(`newdomain.notused')
FEATURE(`masquerade_envelope')
FEATURE(`allmasquerade')
FEATURE(`no_local_masquerading')
MAILER(`smtp')
MAILER(`procmail')

```

**Tabelle 176: m4-Makros**

<b>Makro</b>	<b>Funktion</b>
define	Definiert ein Makro mit einem Argument
undefine	Hebt ein zuvor definiertes Makro wieder auf.
include	Inkludiert die als Argument angegebene Datei.
dn1	Kommentar – Es werden alle Zeichen bis zur neuen Zeile ignoriert.
divert	Steuert die Ausgabe.
feature	Wird verwendet, um neue Funktionen zu aktivieren.

Ein Masterfile wie es z. B. `linux.mc` darstellt, kann mit dem m4-Macroprozessor in eine `sendmail.cf`-Datei übergeführt werden. Man gibt auf der Kommandozeile Folgendes ein:

```
#m4 linux.mc > /etc/sendmail.cf
```

**Achtung**

Seien Sie aber vorsichtig, denn so überschreiben Sie Ihre alte `sendmail.cf`-Datei.

Mit `divert(-1)` erreicht man, dass unnötige Kommentare, in der `sendmail.cf`-Datei nicht mehr vorkommen.

Mit dem Eintrag `FEATURE('always_add_domain')` wird erreicht, dass die Domain immer an Adressen angehängt wird.

Mit dem Eintrag `MAILER('smtp')` wird die SMTP-Mail-Server-Funktionalität eingeschaltet, und mit `MAILER('procmail')` wird ein Interface zu **procmail** geöffnet.

Mit einem Eintrag der Form `define('confAUTO_REBUILD')` wird der Server angewiesen, die Alias-Datenbank immer neu aufzubauen.

Die Alias-Datenbank wird von der Datei `/etc/aliases` aufgebaut. Welchem Zweck dienen die Aliases? Damit der Mail-Server benutzt werden kann, benötigt der Benutzer einen gültigen Account auf dem Server. Die Account-Namen fallen unter die Organisations-Bestimmungen des jeweiligen Unternehmens, z. B. kann ein Account `helmut` heißen oder aber auch `A3q4V5z`. Das

Mail-System arbeitet nun so, dass der Benutzer-Account automatisch die E-Mail-Adresse ist. Wenn wir noch einmal kurz unserer Domain powerup.at treu bleiben, wären damit auch gültige E-Mail-Adresse mit helmut@powerup.at für den User helmut, aber auch A3q4V5z@powerup.at für den User A3q4V5z gegeben. Dies ist natürlich nicht sehr „schön“. Erstens geben wir sofort unseren Maschinen-Account preis, und zweitens ist A3q4V5z auch nicht wirklich leicht zu merken. Uns wäre es viel sympathischer, wenn wir z. B. eine E-Mail Adress helmut.pils@powerup.at hätten.

Genau dies kann der Alias-Mechanismus des Mail-Servers für uns vollbringen. Wir müssen in der Datei /etc/aliases nur unseren gewünschten Alias, gefolgt von der Adresse, die er repräsentiert, aufführen:

```
# This is the aliases file - it says who gets mail for whom.

# Basic system aliases that MUST be present.
postmaster: root
mailer-daemon: postmaster

# amavis
virusalert: root

# General redirections for pseudo accounts in /etc/passwd.
administrator: root
daemon:      root
lp:          root
news:        root
uucp:        root
games:       root
man:         root
at:          root
postgres:    root
mdom:        root
amanda:      root
ftp:         root
wwwrun:      root
squid:       root
mysql:       root
gnats:       root
nobody:      root
# "bin" used to be in /etc/passwd
bin:         root
```



Wenn wir einen Account mit dem Namen A3q4V5z haben und der Benutzer eine E-Mail-Adresse mit `helmut.pils@powerup.at` besitzen, dann schreiben wir in die Alias-Datei die Zeile:

```
helmut.pils:      A3q4V5z
```

Anschließend muss die Alias-Datenbank neu aufgebaut und eingelesen werden. Dies erreicht man durch den Befehl **newaliases**. Nach der Eingabe von `newaliases` kann der Mail-Server E-Mails mit der Adresse `helmut.pils@powerup.at` empfangen und schreibt sie in das Postfach des Users A3q4V5z. Er hängt diese Mails an die Datei `/var/spool/mail/A3q4V5z` an.

Natürlich kann man diesen Mechanismus auch verwenden, um die Mail an mehrere User weiterzuleiten. Wir müssen nur die Namen kommagetrennt angeben. Es sind aber auch andere E-Mail-Adressen erlaubt. Wir könnten z. B. diese Mails an unsere private E-Mail-Adresse weiterleiten lassen. Dazu müssen wir die Zeile in der alias-Datei folgendermaßen abändern:

```
helmut.pils:      A3q4V5z, fuzi@aon.at
```

Damit würde die Mail an `helmut.pils@powerup.at` nicht nur der Benutzer A3q4V5z auf dem lokalen Rechner bekommen, sondern auch an die E-Mail Adresse `fuzi@aon.at` weitergeleitet werden.

Es gibt aber auch noch einen anderen Weiterleitungsmechanismus, den der Benutzer selbst steuern kann. Dies ist oftmals hilfreich, wenn man z. B. auf Urlaub ist und die Mails an das SMS-Gateway weiterleiten lassen möchte. Damit ist man auch im Urlaub immer auf dem Laufenden. Dazu muss der Benutzer in seinem HOME-Verzeichnis eine Datei mit dem Namen **forward** erstellen, deren Inhalt nichts anderes ist, als eine Liste von E-Mail-Adressen, an die die eingehenden E-Mails weitergeleitet werden sollen.

Es könnte der Inhalt einer `.forward`-Datei für den Benutzer A3q4V5z so aussehen:

```
sms-gate@tele.net
```

#### *Hinweis*

Sendmail überprüft allerdings, ob die Rechte dieser Datei so gesetzt sind, dass nur der Eigentümer der Datei Schreib-Rechte besitzt. Dies ist ein Sicherheitsfeature. Ist dies nicht erfüllt, wird die forward-Funktion nicht ausgeführt.

Zu guter Letzt müssen wir unseren DNS-Server von vorhin so anpassen, dass er die Frage nach dem zuständigen Mail-Server für unsere Domain `powerup.at` beantworten kann. Dies wird mit

dem **MX**-Eintrag für **Mail Exchange** erreichen. Wir fügen also in den Zonen-Dateien folgenden Eintrag hinzu:

```
$TTL 1W
```

```
@      IN SOA  tux.powerup.at  root.tux.powerup.at (
        42    ; serial (d. adams)
        2D    ; refresh
        4H    ; retry
        6W    ; expiry
        1W )   ; minimum
```

```
      IN NS   tux.powerup.at
      IN MX 10 mail.powerup.at
```

```
tux  IN  A  172.26.24.23
www  IN  CNAME tux
mail IN  CNAME tux
```

Wir starten den DNS-Server über die Start-Skripte noch einmal neu, und schon sind wir bereit. Unser Mail-Server ist voll einsatzbereit. Wir probieren dies gleich aus, indem wir auf der Kommandozeile das **mail**-Programm verwenden.

Das mail-Programm ist ein zeichenorientiertes Programm, das zum Mailversand und auch praktischerweise aus einem Shell-Skript heraus aufgerufen werden kann. Wir geben Folgendes auf der Kommandozeile ein:

```
#mail -s "Das ist eine Testmail" peter@gmxnet.net
```

Nun verschwindet der Prompt, und wir geben unseren Text ein:

Hallo Peter,

wie geht es dir. Mir geht es gut. Leider habe ich keine Zeit.

Bis dann

Lg

Fredi

.

Der einzelne Punkt ist sehr wichtig, denn dieser kennzeichnet das Ende der Mail. Ein anschließendes Return bringt **EOT** (End of Text) auf den Bildschirm und die Mail wird versandt.

Ob der Versand funktioniert hat, können wir mit dem Befehl **mailq** überprüfen. Das Kommando **mailq** listet alle Mails auf, die in der Mail-Queue auf die Auslieferung warten. Wenn wir hier keinen Eintrag mehr finden, dann haben wir bereits gewonnen, denn, wenn nur wir die einzigen sind, die bei funktionierender Verbindung eine Mail senden, dann ist der Transport der Mail sicher schneller als wir beim Tippen von **mailq**. Die Option **-v** für den **mailq**-Befehl liefert noch mehr Informationen, wenn eine gespoolte Mail in der Queue ist.

Das Programm **mail** kann man nicht nur verwenden, um an entfernte Personen mails zu schicken, sondern man kann diesen Mechanismus für lokale User, unabhängig von einem gestarteten und richtig konfigurierten Mail-Server, immer verwenden.

Wir könnten demnach, auch ohne einen funktionierenden Mail-Server, jedem lokalen User eine Mail schreiben. So wäre es möglich, als **root**-User dem User **helmut** mit folgenden Zeilen eine Mail zu schreiben:

```
#mail -s "Urlaub" helmut
Hallo Helmut wann gehst du auf Urlaub?
.
EOT
#
```

Auch hier wird die Mail an die Datei **/var/spool/mail/helmut** angehängt.

## 14.3 Die große weite Welt – der Web-Server

Das Mitteilungsbedürfnis wird immer größer, das Internet als Werbe- und Präsentationsplattform ist quasi schon etabliert. Aus diesem Grund wenden wir uns jetzt dem Kommunikator, dem Web-Server zu. In der Hitliste der eingesetzten Web-Server führt der Apache Web-Server unangefochten mit ca 63%<sup>6</sup>. Aus diesem Grund werden wir uns auch diesem Web-Server zuwenden und versuchen, ihn zu konfigurieren.

---

<sup>6</sup> Gültiger Wert bei Erstellung des Buches

Bei den meisten Distributionen wird der Apache Web-Server bereits standardmäßig mitinstalliert. Leider werden die Installationsverzeichnisse oft den Regeln der jeweiligen Distribution angepasst. So kann man sich nicht sicher sein, wo die Dateien, die man zur Arbeit benötigt, wirklich liegen. Der Standardpfad wäre **/usr/local/httpd**. Das letzte httpd ist bereits entlehnt vom eigentlichen Servernamen **httpd**. Dies beinhaltet das Protokoll, das der Web-Server spricht, nämlich **http** (Hypertext Transfer Protocol) und das **d** für Daemon.

Wenn das Verzeichnis **/usr/local/httpd/** nicht existiert, dann haben Sie eine Distribution, die den Standardpfad aufteilt. Das macht aber alles nichts, denn wir werden die wichtigsten Dateien kennen lernen und können mit unserem bisher erlangten Wissen auch danach suchen. Wir benötigen zuerst die Konfigurationsdatei **httpd.conf**. Bei der SuSE 9.0 Prof.-Version finden Sie diese Datei im Verzeichnis **/etc/httpd/httpd.conf**.

Wenn wir einen Blick in diese Datei riskieren, werden wir wieder mit einer Fülle von Einträgen förmlich erschlagen. Keine Panik! Wir werden uns nur ein paar ansehen, und Sie werden sehen, eigentlich braucht man gar nichts machen, damit der Web-Server arbeitet.

Zuerst erkennen wir, dass auch diese Konfigurationsdatei in Sektionen, also in Abschnitte unterteilt ist. Dies resultiert aus der geschichtlichen Entwicklung des Apache Web-Servers. Lange Zeit waren drei Konfigurationsdateien anzupassen. Diese drei wurden nun in einer Datei zusammengefasst. Aus diesem Grund sind Abschnitte vorhanden, und die Größe der Datei wirkt oft erdrückend. Allerdings erkennt man bei genauer Betrachtung, dass die meisten Einträge nur Kommentare (**#** Kommentarzeichen) sind. Man könnte alle Kommentare entfernen und hätte damit eine weitaus kleinere Datei vor sich. Ein guter Tipp von mir - lassen Sie die Kommentare darin. Sie werden nicht jeden Tag einen Web-Server konfigurieren, und mit der Zeit vergessen Sie die Einstellungsmöglichkeiten und die Parameter, Sie werden glücklich sein, wenn noch Kommentare mit Beispielen vorhanden sind. Dieser Tipp gilt grundsätzlich.

Sehen wir uns ein paar wichtige Einstellungen an:

ServerType standalone

Mit dieser Einstellung (bei Apache heißen sie **Directives**) kann man festlegen, ob der Web-Server als Daemon über die Start-

Skripte oder ob er über den Internet Super Daemon inetd oder xinetd gestartet wird.

```
ServerRoot "/srv/www"
```

Dieser Eintrag legt die Wurzel des Server fest.

```
Timeout 300
```

Dieser Eintrag bestimmt die Wartezeit in Sekunden, bis beim Laden und Senden ein Timeout erkannt wird.

```
MaxClients 150
```

Damit kann man die Anzahl der gleichzeitigen Client-Anfragen begrenzen. Dies kann z. B. aus Performancegründen notwendig werden.

```
#Listen 3000
```

```
#Listen 12.34.56.78:80
```

Hiermit könnten wir den Port, auf den der Web-Server standardmäßig horcht, überschreiben. Wenn wir also bei

```
#Listen 3000
```

das Kommentarzeichen # löschen, würde unser Web-Server nicht auf den Standardport 80 lauschen, sondern auf 3000. Dies kann notwendig sein, wenn man als unprivilegierter User einen Web-Server betreiben will. Denn nur der SuperUser root hat das Recht, Server zu starten, die auf Ports hören die kleiner als 1024 (die privilegierten Ports) sind.

```
LoadModule  mmap_static_module  /usr/lib/apache/mod_mmap_static.so
LoadModule  vhost_alias_module  /usr/lib/apache/mod_vhost_alias.so
LoadModule  env_module          /usr/lib/apache/mod_env.so
LoadModule  define_module       /usr/lib/apache/mod_define.so
LoadModule  config_log_module   /usr/lib/apache/mod_log_config.so
LoadModule  agent_log_module    /usr/lib/apache/mod_log_agent.so
LoadModule  referer_log_module  /usr/lib/apache/mod_log_referer.so
```

```
. . .
. . .
```

In diesem Bereich kann man alle Module, die die Funktionalität des Web-Servers erweitern, laden. Dies ist ein herausragendes Konzept. Nur das, was man wirklich für den reibungslosen Betrieb benötigt, wird als Modul geladen. Damit bleibt der eigentliche Dienst immer so klein und schnell wie möglich.

Wir haben schon gehört, dass nur der SuperUser `root` das Recht hat, Server zu starten, die auf Ports hören, die kleiner als 1024 sind. Nun wäre es ein großes Sicherheitsrisiko, wenn der Web-Server mit `root`-Rechten laufen würde. Die Apache Entwickler haben sich deshalb etwas Geniales einfallen lassen. Nur beim Start wird ein Prozess erzeugt, der unter `root`-Rechten läuft, und einige andere, die einem unprivilegierten Benutzer gehören. Diese sind die eigentlichen Arbeitstiere. Denn der `root`-Prozess leitet die Anfragen einfach an diese unprivilegierten weiter.

```
User wwwrun
```

```
Group www
```

Mit diesen Einstellungen kann man den Benutzer und die Gruppe des unprivilegierten Benutzers einstellen, unter der der eigentliche Apache laufen soll.

```
DocumentRoot "/srv/www/htdocs"
```

Dies ist nun der Eintrag, unter dem wir im Betrieb die eigentlichen HTML-Dokumente finden und abspeichern werden. Dies ist der `/` in der URL von z. B. `www.powerup.at`. Wenn wir dies im Browser eingeben, holt der Web-Server die HTML-Dokumente (standardmäßig ***index.html***) aus dem lokalen Verzeichnis ***/srv/www/htdocs***.

Nun kommen die sogenannten ***Directory Container***. Für jedes Directory, in dem man HTML-Dokumente unterbringen möchte, benötigt man einen Directory Container, der beschreibt, wie sich das Verzeichnis verhalten soll.

```
<Directory "/srv/www/htdocs">
```

```
Options Indexes -FollowSymLinks +Includes
```

```
AllowOverride None
```

```
Order allow,deny
```

```
Allow from all
```

```
</Directory>
```

Bei diesem Directory Container bestimmen wir mit dem Optionseintrag, dass, wenn die Datei `index.html` (also jene Datei, die konfiguriert wurde) nicht existiert, ein Verzeichnisindex angezeigt wird. Also der gesamte Inhalt des Verzeichnisses. Dies ist aus sicherheitstechnischen Gründen allerdings nicht zu empfehlen, deshalb sollte das Argument `Indexes` aus der Optionszeile entfernt werden. Das `-FollowSymLinks` verhindert, dass anstelle der echten Datei ein symbolischer Link steht und die Zieldatei nicht unterhalb der Web-Server-Wurzel liegt.

Der Eintrag `AllowOverride` wird benötigt, um festzuhalten, ob die Optionen überschreibbar sind oder nicht. Bei `Allow from all` kann man nun einzelne Domains oder IP-Adressen angeben, für die dieses Verzeichnis einsehbar ist. Wenn wir z. B. Folgendes hier stehen hätten:

```
Allow from 172.26.24.55
```

Dann würde nur der Rechner mit der IP-Adresse 172.26.24.55 das Recht haben, die Seiten aus diesem Verzeichnis anzusehen. Der Directory Container wird nun wieder mit einem `<html>` erinnernden Tag geschlossen.

Wir stoßen auch auf Einträge der Form:

```
AddType application/x-httpd-php .php4
```

Damit werden so genannte **MIME**-Typen definiert. MIME heißt Multipurpose Internet Mail Extensions und wird dazu benötigt, dem Web-Server und dem Browser mitzuteilen, was und wie er die folgenden Daten interpretieren soll. Wenn wir z. B. den obigen Eintrag in unserer Konfigurationsdatei nicht hätten, würden alle über **PHP** (eine Skriptsprache) erstellten dynamischen HTML-Seiten nicht richtig angezeigt, sondern anstelle des Ergebnisses einfach nur der Sourcecode.

#### *Hinweis*

Der KDE arbeitet bei seinen Icons auch mit den MIME-Typen.

Wenn wir diese Anpassungen vorgenommen haben bzw. wenn wir wissen, wo wir unsere HTML-Dokumente hinspeichern müssen, um Sie von unserem Web-Server aus verteilen zu lassen, können wir den Web-Server starten. Dies erfolgt entweder über das Startkommando ***apachectl*** (Apache Control) mit

```
#apachectl start
```

oder einfach wieder über ein Start-Skript **`/etc/init.d/httpd start`**. Wenn wir wollen, dass der Server bei jedem Neustart automatisch gestartet wird, müssen wir wiederum einen Link in das entsprechende Verzeichnis in der richtigen Form machen.

### Übung

Führen Sie alle Tätigkeiten durch, um den Web-Server bei einem Reboot in den Runlevel 5 automatisch zu starten.

### Übung

Damit wir unseren Web-Server auch über den Namen `www.powerup.at` aufrufen können, benötigen wir einen Eintrag in den DNS-Server. Führen Sie bitte alle nötigen Arbeitsschritte aus, damit Sie in Ihrem Browserfenster mit `www.powerup.at` die richtigen Seiten angezeigt bekommen.



Wenn wir in einem Netzwerk arbeiten, wollen wir natürlich mit den Kollegen Dateien austauschen und dies, wenn es geht, auch komfortabler als mit dem FTP-Dienst. Unter Windows kennen wir die Netzwerkumgebung, die mit den **SMB** (Server Message Block)-Protokoll arbeitet. Unter Linux haben wir schon öfters vom **NFS** (Network File System)-Dienst gesprochen, der das Gegenstück der Windows-Freigaben auf Linux-Seite darstellt. Wir wollen uns nun im Folgenden die Dienste NFS und **Samba** ansehen. Mit dem Samba-Dienst kann man eine Brücke zwischen der Linux-Welt und der Windows-Welt schlagen.

### 15.1

#### Der NFS-Dienst

Das NFS wurde entwickelt, um es in einem Netzwerk möglich zu machen, ein freigegebenes Verzeichnis eines beliebigen Computers auf einen anderen Computer zu mounten, als ob es sich dabei um ein lokales Laufwerk handeln würde. Natürlich gibt es auch Vorkehrungen, damit dies nur bestimmten Personen auf bestimmten Computern möglich ist – aus Sicherheit.

Der NFS-Dienst funktioniert mittels UDP-Protokoll und damit verbindungslos, aber extrem schnell. Weiter benutzt NFS die sogenannten **Remote Procedure Calls (rpc)** und deshalb findet man das `rpc` in allen Namen der benötigten Prozesse. Wir benötigen den eigentlichen NFS-Serverprozess, der heißt **`rpc.nfsd`**. Wir benötigen einen Dienst, der die Mountanfragen bearbeitet – dieser heißt **`rpc.mountd`**. Des Weiteren benötigen wir noch den **Portmapper**-Dienst, der die Port-Nummern zu `rpc`-Prozessen mapped. Alle benötigten Dienste können wieder über die Start-Skripte im Verzeichnis `/etc/init.d` gestartet werden. Das Start-Skript hat üblicherweise den Namen **`nfsserver`**. Sie brauchen jetzt keine Angst bekommen, dass jeder Dienst eine eigene Konfigurationsdatei besitzt, nein, wir müssen nur eine einzige Datei unseren Bedürfnissen anpassen, und diese heißt **`/etc/exports`**.

Dieser Name verrät uns auch gleich, was wir darin konfigurieren werden – nämlich Freigaben bzw. exportierte Verzeichnisse.

Ein Eintrag in der Datei `/etc/exports` sieht üblicherweise so aus:

```
Verzeichnis maschine1(option11,option12) maschine2(option21,option22)
```

Hierbei bezeichnet Verzeichnis das freizugebende Directory und maschine1 und maschine2 die Client-Rechner, die Zugriff auf das Verzeichnis erhalten sollen. Bei den Maschinennamen kann man noch Optionen angeben, die die Art und Weise definieren, wie diese Clients auf das Verzeichnis zugreifen dürfen.

**Tabelle 177: Optionen für NFS-Freigaben**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
ro	Read Only. Der Clientrechner hat nur Lese-Rechte – dies ist die Defaulteinstellung.
rw	Read Write. Nun darf der Client sowohl lesen als auch schreiben.
no_root_squash	Normalerweise werden die Anfragen des root-Users auf der Clientmaschine auf dem Server so behandelt, als wenn sie von lokalen User nobody oder oft auch nfsnobody kommen würden. Mit dieser Option kann man den Server so einstellen, dass er root-Clientanfragen mit root-Rechten am Server gestattet. Bitte mit Vorsicht genießen, denn damit kann man ernste Sicherheitsprobleme bekommen.
anonuid und anongid	Diese Optionen setzen explizit die UID und die GID des anonymen Accounts. Diese Optionen sind dann nützlich, wenn man möchte, dass alle Anfragen von einem Client kommend erscheinen.

all_squash	Übersetzt alle UID und GID auf den anonymen Account.
sync	Erzwingt, dass alle Schreiboperationen auf der Platte abgeschlossen sein müssen, bevor die Anfrage beendet werden kann. Dies ist die Defaulteinstellung.
async	Mit dieser Option erreicht man einen Performancegewinn, da die Voraussetzungen, wie sie sync fordert, nicht erfüllt werden müssen. Allerdings kann es im Fehlerfall zu Datenverlusten führen, da keine Kontrolle über die korrekte Schreibtätigkeit vorhanden ist.

Wenn man nun Änderungen an der Datei `/etc/exports` vorgenommen hat, muss man mit dem Befehl **exportfs** und der Option `-a` diese Änderungen auch aktivieren.

Mit dem Befehl `exportfs` kann man auch individuell und schnell Verzeichnisse exportieren, ohne dass man die Datei `/etc/exports` editiert. Allerdings sind diese Freigaben nur temporär und nach dem nächsten Booten wieder verloren.

```
#exportfs -o no_root_squash helmut:/www
```

Dieser Befehl gibt das Verzeichnis `/www` für den Rechner `helmut` mit der Option `no_root_squash` frei.

**Tabelle 178: Optionen für exportfs**

<b>Option</b>	<b>Bedeutung</b>
<code>-a</code>	Exportiert oder sperrt alle Verzeichnisse.
<code>-o</code>	Optionen, wie in der Datei <code>exports</code> verwendet.
<code>-i</code>	Ignoriert die Datei <code>/etc/exports</code> .
<code>-r</code>	Reexportiert wieder alle Verzeichnisse.

-u	Sperrt ein oder mehrere Verzeichnisse wieder.
-v	Verbose.

Wenn man freigegebene Verzeichnisse in einem Netzwerk hat, wollen wir sie auch benutzen, also auf unserem System je nach Freigabe-Rechten mounten.

#### *Hinweis*

Es prallen hier zwei Rechtssysteme aufeinander. Das eine ist das Rechtssystem des Dateisystems, und das andere ist das Rechtssystem der Freigabe. Das bedeutet, wenn ich auch bei der Freigabe alles erlaube, aber auf dem Dateisystem stimmen die Rechte nicht mit denen der Freigabe überein, funktioniert es nicht. Man kann sagen, das letzte Wort hat weiterhin das Dateisystem, und das ist gut so. Bei Windows-Systemen ist es genauso. Hier wird auch zwischen Freigabe-Rechten und Dateisystem-Rechten unterschieden.

Der mount-Befehl, wie wir ihn kennen, wird einfach um eine Komponente erweitert und zwar um die Angabe eines Servers.

```
mount server:/Ver/zeich/nis /mountpoint
```

Wobei server nicht notwendigerweise ein Name sein muss, sondern natürlich auch eine IP-Adresse sein kann. Nehmen wir an, wir hätten einen Rechner bei uns im Netzwerk, der helmut heißt. Auf diesem Rechner ist ein Verzeichnis mit dem Namen /ftp/pub freigegeben. Dieses Verzeichnis wollen wir in unsere Verzeichnishierarchie einbinden und zwar im Unterverzeichnis /mnt/remote. Die Befehlszeile dazu sieht folgendermaßen aus:

```
#mount helmut:/ftp/pub /mnt/remote
```

Wenn wir wollen, dass auf anderen Rechnern freigegebene Verzeichnisse bei jedem Neustart unseres Clients wieder aufs Neue eingehängt werden, müssen wir diese in der Datei **/etc/fstab** spezifizieren. Wir verwenden den selben Mechanismus beim automatischen Einhängen, als ob dies lokale Laufwerke wären.

Wir kennen die Datei /etc/fstab bereits und wiederholen an dieser Stelle die möglichen Optionen für diese Datei.

**Tabelle 179: Optionen für die Datei fstab**

<b>Option</b>	<b>Bedeutung</b>
rw	Mountet das Verzeichnis mit Lese- und Schreib-Rechten.
ro	Das Verzeichnis bzw. das Dateisystem (nfs) wird read only gemounted.
bg	Wenn das Mounten fehlgeschlagen ist, wird es im Hintergrund weiter probiert.
hard	(Timeoutoperation) Wenn ein NFS-Server aus irgendwelchen Gründen nicht mehr antwortet, wird auf der Console „Serv not responding“ ausgegeben und unendlich lange probiert, bis er wieder antwortet. Dies ist die Defaulteinstellung.
soft	(Timeoutoperation) Wenn ein NFS-Server aus irgendwelchen Gründen nicht mehr antwortet, dann wird an die Applikation ein I/O-Error gesendet, und es wird unendlich weiter probiert
intr	Erlaubt es einem User, aus einer der obigen Timeout-Operationen auszusteigen.
retrans=NUM	Anzahl der Timeouts, bevor ein hard error zurückgegeben wird.
retry=NUM	Anzahl der Mountversuche (im Vordergrund oder im Hintergrund), bevor aufgegeben wird.

So könnten wir folgende Zeile in unserer Datei /etc/fstab hinzufügen:

```
helmut:/ftp/pub /mnt/remote nfs rw 0 0
```

Damit würde bei jedem Reboot des Rechners versucht, das Verzeichnis /ftp/pub vom Rechner helmut bei mir local in das Verzeichnis /mnt/remote einzuhängen.

Wenn wir an statistischen Informationen über unseren NFS-Server interessiert sind, können wir diese mit dem Befehl **nfsstat** erhalten.

**Tabelle 180: Optionen für nfsstat**

<b>Option</b>	<b>Bedeutung</b>
-s	Gibt nur Statistiken den Server betreffend aus.
-c	Gleiches wie -s, nur Clients betreffend.
-r	Zeigt nur rpc-Statistiken an.
-n	Nur nfs-Informationen werden angezeigt.
-z	Die Statistikeinträge werden wieder alle auf Null zurückgesetzt.
-o FAC	Zeigt nur FAC Informationen an. FAC können sein: nfs - NFS-Protokoll-Informationen. rpc - RPC-Informationen. net - Netzwerkschicht-Informationen. fh - Datei-Handle-Informationen.

Abschließend noch eine Übersicht über die verwendeten Port-Nummern beim NFS-Dienst:

**Tabelle 181: Port-Nummer und Dienste**

<b>Dienst</b>	<b>Protokoll</b>	<b>Port-Nummer</b>
nfsd	TCP und UDP	2049
mountd	UDP	745
mountd	TCP	747
portmapper	TCP und UDP	111

## 15.2

## Samba – das Fenster zu Windows

Bei Samba handelt es sich nicht um den allseits beliebten Tanz, sondern um die freie Implementierung des Server Message Block Protokolls, das von den Windows-Rechnern verwendet wird, um mit Freigaben umzugehen. Samba ist somit zu einem sehr populären und beliebten Server geworden, denn in den meisten Fällen stehen Linux-Rechner in einer heterogenen Umgebung, wo Windows und Linux gemeinsam vorkommen. Reine Monokulturen an Betriebssystemen sind eher die Ausnahme. Der Samba-Dienst setzt sich aus zwei Daemon-Prozessen zusammen, dem **smbd** (Server Message Block Daemon), der für die Datei und Druckfreigaben zuständig ist, und dem **nmdbd** (Name Message Block Daemon), der für die Namensauflösung verantwortlich zeichnet.

Der Samba-Dienst kann wieder mit einem Start-Skript mit Namen **/etc/init.d/smb** gestartet und gestoppt werden. Wenn ein automatischer Start gefordert ist, dann muss man wieder einen Link in den passenden Runlevel erzeugen. Die Namenskonvention nicht vergessen!

Der Samba-Server wird über eine große Konfigurationsdatei mit dem Namen **smb.conf** eingestellt. Diese findet man in **/etc/samba/smb.conf** oder in **/etc/smb.conf**.

```
# smb.conf is the main Samba configuration file. You find a
# full #commented
#                               version                               at
/usr/share/doc/packages/samba/examples/smb.conf.SuSE
# Date: 2003-09-23
[global]
    workgroup = HOME
    os level = 2
    time server = Yes
    unix extensions = Yes
    encrypt passwords = Yes
    map to guest = Bad User
    printing = CUPS
    printcap name = CUPS
    socket options = SO_KEEPALIVE IPTOS_LOWDELAY TCP_NODELAY
    wins support = No
    veto files = /*.eml/*.nws/riched20.dll/*.{*}/
[homes]
    comment = Home Directories
    valid users = %S
```

```

browseable = No
read only = No
create mask = 0640
directory mask = 0750
[printers]
comment = All Printers
path = /var/tmp
printable = Yes
create mask = 0600
browseable = No
[print$]
comment = Printer Drivers
path = /var/lib/samba/drivers
write list = @ntadmin root
force group = ntadmin
create mask = 0664
directory mask = 0775

```

Diese Datei setzt sich wieder aus Sektionen zusammen, wobei wir in Globale-Sektion und in Freigabe-Sektion (Share-Sektion) unterscheiden wollen.

In der Globalen-Sektion wird alles eingestellt, was den gesamten Server und dessen Verhalten am Netzwerk betrifft. Wir finden hier Einträge über die Art und Weise, wie die Anmeldung funktionieren soll, welche Art von Passwörtern er akzeptieren wird, ob es sich um eine Workgroup oder einen Domänen-Controller handelt und vieles mehr. In der Share-Sektion können nur Einträge gefunden werden, die sich auf die Freigabe der Verzeichnisse und deren Eigenschaften beziehen.

**Tabelle 182: Optionen für die Datei smb.conf**

<b><i>Eintrag</i></b>	<b><i>Funktion</i></b>
workgroup=NAME	Name der Arbeitsgruppe oder der NT-Domäne.
server string = NAME	Name, der in der Netzwerkumgebung angezeigt wird.
hosts allow = NAME	Eine Liste von Rechnern, die sich mit dem Server verbinden dürfen.



security= TYP	Hier sind für TYP folgende Einträge möglich: share, user, server und domain. Bei neueren Windowsumgebungen sollte die Einstellung auf user lauten.
encrypt passwords=yes no	Die Passwörter werden verschlüsselt. Dies muss für die neueren Versionen von Windows aktiviert werden. (NT, 2000, ...).
Interfaces=NAME	Bestimmt, welches Netzwerkinterface benutzt werden soll.
domain master = yes no	Definiert, dass der Linux-Rechner sich wie ein NT-Domain-Controller verhält.
wins support = yes no	Aktiviert oder deaktiviert die Unterstützung der WINS-Funktionalität.

Eine Share-Definition könnte folgendermaßen aussehen.

[Briefe]

```
comment = Briefe privat
browsable = yes
writeable = yes
path = /docs/briefe
create mask = 0640
directory mask = 0750
```

Mit der Option browsable=yes erreichen wir, dass dieses Verzeichnis in der Netzwerkumgebung angezeigt wird. Der Pfad definiert das freigegebene Verzeichnis, und mit writeable erlauben wir auch das Schreiben in dieses Verzeichnis. Mit den mask-Angaben definieren wir die Standardrechte, mit denen Dateien und Verzeichnisse in diesem Verzeichnis angelegt werden.

Wenn wir mit allen unseren Einstellungen an der smb.conf fertig sind, können wir die Korrektheit der Einträge mit dem Befehl **testparm** überprüfen lassen.

```
#testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Processing section "[print$]"
Loaded services file OK.
Press enter to see a dump of your service definitions
# Global parameters
[global]
    coding system =
    client code page = 850
    code page directory = /usr/share/samba/codepages
    workgroup = HOME
    netBIOS name =
    netBIOS aliases =
    netBIOS scope =
    server string = Samba 2.2.8a-SuSE
    interfaces =
    bind interfaces only = No
    security = USER
    encrypt passwords = Yes
    update encrypted = No
    allow trusted domains = Yes
    hosts equiv =
    min passwd length = 5
    map to guest = Bad User
    null passwords = No
    obey pam restrictions = No
    password server =
    smb passwd file = /etc/samba/smbpasswd
    . . .
    . . .
```

### *Achtung*

Das Kommando `testparm` überprüft nur die korrekte Syntax. Es liefert keine Aussagen über die korrekte Funktionsweise des Servers.

Wir können unseren Samba-Server wieder über die Start-Skripte starten. Mit dem Befehl **`smbstatus`** können wir den Status unseres Samba-Servers überprüfen.

Nun, da wir einen Server laufen haben, können sich die User in unserem Netzwerk schon mit dem Server verbinden. Da aber der Security-Eintrag auf user stehen sollte, müssen wir noch diejenigen User anlegen, die mit dem Server kommunizieren dürfen. Der Samba-Server setzt als User einen Benutzer voraus, der auf der Maschine einen gültigen Account, also einen Linux-Account besitzt. Erst dann kann man den Benutzer auch als Samba-Benutzer definieren. Dies funktioniert mit dem Befehl **smbpasswd**. Wenn wir einen Benutzeraccount `helmut` auf unserem Server haben, kann dieser Benutzer `helmut` auch als Samba-Server-Benutzer definiert werden.

```
#smbpasswd -a helmut
```

New SMB password:

Retype new SMB password:

Password changed for user helmut:

Password changed for user helmut:

Mit der Option `-a` haben wir den Samba-Server angewiesen, den User `helmut` in die Samba-eigene Passwortdatei aufzunehmen und ihm gleichzeitig ein Passwort zu verpassen. Ohne die Option `-a` würde lediglich das Passwort eines bereits bestehenden Benutzers geändert.

Nun ist alles „roger“ für den ersten Login. Da der User `helmut` einen UNIX-Account auf der Maschine besitzt, hat er auch ein HOME-Verzeichnis. Dieses HOME-Verzeichniss ist immer freigegeben, aber nicht browsable. Also jeder User hat immer sein persönliches HOME-Verzeichnis auch über den Samba-Server zur Verfügung.

Drehen wir den Spieß jetzt einmal um: nicht unser Server soll die Windows-Clients versorgen, sondern ein Windows-Server soll unsere Linux-Clients versorgen. Dazu verwenden wir den Befehl **smbclient**. Der Befehl `smbclient` ist ein Kommandozeilen-orientierter Befehl. Wenn wir z. B. überprüfen wollen, welche Ordner auf einem Windows-Rechner mit dem Namen `buch`, freigegeben sind, verwenden wir folgenden Befehl:

```
#smbclient -L buch -U Administrator
```

```
added          interface          ip=172.26.24.23          bcast=172.26.24.255
nmask=255.255.255.0
```

Got a positive name query response from 172.26.24.111 ( 172.26.24.111 )

Domain=[LINUXBUCH] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]

Sharename	Type	Comment
-----	----	-----
IPC\$	IPC	Remote-IPC
docs	Disk	
ADMIN\$	Disk	Remoteadmin
C\$	Disk	Standardfreigabe
doks	Disk	

Server	Comment
-----	-----
BUCH	

Workgroup	Master
-----	-----
HOME	MAXI
LINUXBUCH	BUCH

In diesem Fall ist buch der NetBIOS-Name. Wir könnten genauso gut nur die IP-Adresse des Rechners angeben, wenn wir z. B. den NetBIOS-Namen nicht kennen.

Die Workgroup heißt LINUXBUCH, und das Betriebssystem ist Windows 2000. Als Freigaben finden wir IPC\$, docs, ADMIN\$, C\$ und doks. Wir wissen, dass IPC\$, ADMIN\$ und C\$ die versteckten (durch das schließende \$ gekennzeichnet) Freigaben von Windows-Rechnern sind, die man nicht oder nur sehr schwer und mit besonderer Sorgfalt abschalten kann.

**Tabelle 183: Optionen für den smbclient**

<i><b>Option</b></i>	<i><b>Bedeutung</b></i>
-s DATEI	Pfadangabe für eine alternative smb.conf.
-M HOST	Sendet ein WinPopUp an den mit HOST spezifizierten Rechner.
-N	Fragt nach keinem Passwort.
-p NUM	Manuelle Spezifikation der zu verwendenden Port-Nummer.

-h	Listet alle möglichen Kommandozeilenparameter auf.
-E	Die Ausgabe von stdout wird auf stderr umgeleitet.
-U NAME	Spezifiziert den Usernamen für die Verbindung.
-L HOST	Listet Informationen über den mit HOST spezifizierten Rechner auf. HOST kann der NetBIOS-Name oder auch eine IP-Adresse sein.
-W NAME	Damit kann die Arbeitsgruppe definiert werden.

Wir wollen als Beispiel mit dem Verzeichnis C\$ auf dem Rechner buch arbeiten.

```
#smbclient //buch/C$ -U Administrator
added interface ip=172.26.24.23 bcast=172.26.2.4.255
nmask=255.255.255.0
Got a positive name query response from 172.26.24.111
Password:
Domain=[LINUXBUCH] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> ls
  AUTOEXEC.BAT           H           0 Tue Jul 22 09:00:47 2003
  backup notebook        D           0 Fri Oct 31 08:06:28 2003
  BJJPrinter              DH          0 Tue Oct 21 17:05:30 2003
  boot.ini                HS          192 Tue Jul 22 09:53:30 2003
  CONFIG.SYS              H           0 Tue Jul 22 09:00:47 2003
  doks                    D           0 Fri Oct 31 09:01:45 2003
  Dokumente und Einstellun DA          0 Tue Jul 22 09:10:39 2003
  gs                       D           0 Sun Sep 14 16:45:13 2003
  IO.SYS                  AHSR        0 Tue Jul 22 09:00:47 2003
  lisi-kinderkurse.pdf    A          90217 Tue Sep 30 19:45:36 2003
  MSDOS.SYS               AHSR        0 Tue Jul 22 09:00:47 2003
  My Documents            D           0 Tue Jul 22 11:54:30 2003
. . .
. . .
  WINNT                   DA          0 Thu Sep 18 18:33:10 2003
  WUTemp                  DA          0 Fri Aug 1 14:51:42 2003

38154 blocks of size 1048576. 34783 blocks available

smb: \> exit
```

Auf dieser Ebene können wir mit dem Befehl **help** erfahren, welche Kommandos das interaktive **smbclient**-Programm verarbeiten kann.

**Tabelle 184: Kommandos im smbclient**

<b>Kommando</b>	<b>Funktion</b>
? COM	Hilfeangaben für das Kommando COM.
! COM	Das Shell-Kommando COM wird ausgeführt.
cd DIR	Wechselt in das Verzeichnis DIR.
del NAME	Löschen.
dir NAME	Zeigt den Inhalt eines Verzeichnisses an.
get	Funktioniert wie beim ftp mit Angabe von remote-Datei und lokaler-Datei.
put	Wie get, nur in die andere Richtung.
lcd	Wechselt lokal das Verzeichnis.
md DIR	Erzeugt ein neues Verzeichnis.
rd DIR	Löscht ein Verzeichnis.
print NAME	Druckt eine lokale Datei NAME auf einen freigegebenen Drucker aus.
tar	Führt tar-Operationen aus.
setmod	Setzt die Dateiattribute. Ist mit dem DOS-Befehl attribute zu vergleichen.

Da der **smbclient** etwas an einen FTP-Client angelehnt ist, kommt man relativ schnell mit ihm klar und kann ganz nett damit arbeiten. Wir wollen die Bequemlichkeit aber auf die Spitze treiben und ein von Windows freigegebenes Verzeichnis bei uns auf dem Linux-Rechner mounten, als ob es ein lokales Laufwerk

wäre – so wie beim NFS-Dienst. Da Linux eine große Anzahl von Dateisystemen unterstützt und eines davon **smbfs** (Server Message Block File System) ist, können wir den mount-Befehl dazu verwenden.

```
#mount -t smbfs -o username=NAME //SERVER/FREIGABE /MOUNTPOINT
```

Wenn – was eigentlich immer sein sollte – ein Passwortschutz vorhanden ist, wird dieser einfach mit angegeben. Das Kommando dazu würde dann so aussehen, wenn wir unsere Freigaben von vorhin wieder verwenden:

```
#mount -t smbfs -o username=Administrator \  
-o password=XYZ //buch/C$ /mnt/NT
```

Jetzt können wir auf die Platte C:\ des Windows-Rechners zugreifen, indem wir in das Verzeichnis /mnt/NT wechseln.

Ich habe lange überlegt, ob dem Bereich Sicherheit ein eigenes Kapitel gewidmet werden soll oder ob bei der Behandlung der einzelnen Komponenten schon die sicherheitstechnischen Bedenken und Risiken aufgeführt werden sollten. Meinen innerlichen Kampf haben Sie – wenn Sie jetzt so zurückblicken – bereits erkannt, denn ich habe bei den einzelnen Komponenten auch schon immer auf die Sicherheitsbedenken Bezug genommen, und dennoch befinden Sie sich in dem eigenen Kapitel Sicherheit.

Wir werden in diesem Kapitel noch einmal die wichtigsten Security-Themen aufgreifen und erläutern und insbesondere das für die Fernadministration so wichtige Tool wie die Secure-Shell besprechen. Den TCP-Wrapper haben wir bereits ausführlich besprochen. Natürlich ist die Einschränkung der Dienste, die über den `inetd` aufgerufen werden können, auf gewisse IP-Adressen ein äußerst wichtiger Beitrag zur Verbesserung der Sicherheit.

## 16.1

### **Sicherheitsrichtlinien – was sollte geprüft werden**

Einige Anwendungen und Programme, die wir unter Linux finden, haben bei den Berechtigungen das SUID oder das SGID-Bit gesetzt. Wenn wir uns erinnern, soll das dazu dienen, dass die Programme von allen Benutzern aufgerufen werden können, aber zum Zeitpunkt der Ausführung unter den Eigentümer- oder Gruppenrechten laufen. Ein Beispiel dafür wäre das `passwd`-Programm. Eigentlich hat nur der SuperUser das Recht, die Passwörter zu verändern. Damit aber ein normaler User sein Passwort jederzeit ändern kann und dieses vielleicht auch muss, ist beim `passwd`-Programm das SUID-Bit gesetzt.

Aber genau dieser Mechanismus des SUID und SGID kann benutzt werden, um auf einem Rechner `root`-Rechte oder höhere Rechte zu erlangen. Aus diesem Grund sollte man regelmäßig das Verzeichnissystem darauf hin überprüfen. Wir können dazu das Bordmittel `find` verwenden.



```
#find / -user root -perm -4000 -ls | mail \
- s "Achtung" admin@rechnet.net
```

Es hat in der Vergangenheit schon öfters bei verschiedenen Distributionen den Fall gegeben, dass SUID oder SGID-Bits bei der Installation automatisch auf Dateien gesetzt wurden, die man nur zum Zeitpunkt der Installation benötigte, und das Installationsprogramm hat dann „vergessen“, diese Bits wieder zu entfernen. Deshalb ist dieser Check auch gleich nach der Installation empfehlenswert.

#### *Hinweis*

Natürlich haben Sie erkannt, dass das obige Kommando nur das SUID-Bit sucht. Sie müssen also auch ein Kommando für das SGID-Bit erstellen und ausführen. Das alles sollte auch noch mittels crond automatisiert werden.

Von den beiden Varianten, wie die Passwörter auf einem Linux-Rechner verwaltet und gespeichert werden, ist immer und ausschließlich das Shadow-System zu bevorzugen. Der Unterschied liegt darin, dass im Shadow-System die verschlüsselten Passwörter in einer eigenen Datei (shadow) in einer nur für den root-User zugänglichen Form gespeichert sind. Bei der ursprünglichen Form sind die verschlüsselten Passwörter weltlesbar in der Datei passwd gespeichert. Des Weiteren kann man die Sicherheit der Passwörter noch durch Verwendung eines neueren Verschlüsselungsalgorithmus erhöhen. Der ursprüngliche Crypt-Algorithmus verschlüsselt Passwörter nur bis zu den ersten 8 signifikanten Stellen. Damit ist ein Passwort zu cracken leichter möglich, als bei der Verwendung von z. B. des **md5**-Algorithmus, der viel mehr Stellen für signifikant erachtet.

Bei der Verwendung des Shadow-Systems muss man natürlich darauf achten, dass die Datei /etc/shadow, die ja die verschlüsselten Passwörter beinhaltet, auch immer die Filepermission von 600 besitzt. Die Datei /etc/passwd hat die Berechtigungen 644.

Beim Shadow-System kann man auch mit dem sogenannten Passwort-aging die Sicherheit zusätzlich erhöhen, indem man für die Dauer der Gültigkeit eines Passwortes z. B. 2 Monate einstellt, und nach Ablauf dieser 2 Monate muss der User sein Passwort ändern. Hier möchte ich raten, behutsam mit diesem Mittel umzugehen. Wenn man die Zeit zwischen dem sicherlich

sehr sinnvollen Passwortwechsel zu kurz ansetzt, werden Ihre User immer wieder dieselben zwei oder drei Passwörter verwenden. Dies ist zwar besser als nur eines, aber führt den beabsichtigten Zweck unseres Tuns ad absurdum. Genauso wichtig wie der Wechsel des Passworts ist die Verwendung eines „starken“ Passworts. Was versteht man aber unter einem starken Passwort? Sicherlich nicht den eigenen Vornamen oder die Namen seiner Kinder oder Haustiere. Diese Passwörter sind mit etwas Recherche leicht zu erraten.

Ein starkes Passwort zeichnet sich dadurch aus, dass es nicht in einem Wörterbuch zu finden ist und dass eine sogenannte BruteForce-Attacke unzumutbar lange dauern würde. Diese Methode führt immer zu einem Erfolg. Es ist nur eine Frage der Zeit. Eine BruteForce-Attacke ist, einfach alle möglichen Zeichenkombinationen aus einem Zeichenvorrat auszuprobieren solange bis man Erfolg hat.

Den Zeichenvorrat bestimmen Sie. Wenn Sie Passwörter nur mit Kleinbuchstaben verwenden, dann ist der Zeichenvorrat 26. Wenn Sie allerdings Groß- und Kleinbuchstaben verwenden, haben Sie immerhin schon einen Zeichenvorrat von 52. Das Beste ist allerdings, wenn Sie aus dem Vollen schöpfen und sich des Zeichenvorrats des gesamten 7-Bit ASCII-Zeichensatzes bedienen, denn damit ist der Zeichenvorrat bereits auf 128 Zeichen angewachsen.

Nun kommt es nur noch auf die gekonnte Kombination der Zeichen in einer vernünftigen Länge an. Die Kombination sollte leicht zu merken, aber nicht leicht erratbar sein. Die Länge sollte um die 8 Stellen sein - je mehr desto sicherer<sup>7</sup>.

In den folgenden zwei Tabellen ist die Zeitdauer zum Entschlüsseln der Passwörter gegenübergestellt (diese Werte beziehen sich auf einen Pentium 4/2,5 GHz)<sup>8</sup>.

---

<sup>7</sup> Gilt nur für die neuen Algorithmen wie md5 etc.

<sup>8</sup> Quelle: Chip; 9/2003; Seite 121

**Tabelle 185: Zeichenvorrat 26 – nur Kleinbuchstaben**

<b>Passwort-Länge</b>	<b>Mögliche Kombinationen</b>	<b>Dauer bis entschlüsselt</b>
4	456.976	18 Millisekunden
8	208.827.064.576	2,3 Stunden
12	95.428.956.661.682.200	121 Jahre

**Tabelle 186: Zeichenvorrat 128 – 7 Bit ASCII**

<b>Länge</b>	<b>Kombinationen</b>	<b>Dauer</b>
4	268.435.456	11 Sekunden
8	72.057.594.037.927.900	91 Jahre
12	19.342.813.113.834.100.000.000.000	24,5 Milliarden Jahre

Mein Tipp für die Wahl eines sicheren Passworts ist, sich einen Satz auszudenken, den man sich leicht merkt – z. B. ein Lebensmotto und diesen mit den Kürzeln der amerikanischen Sprechweise und der EDV zu „codieren“.

Die Kombination **2b!2b?** wäre schon ein ganz gutes Passwort – bis auf die Länge. Wie soll man sich das merken, meinen Sie? Ganz einfach. Mit den gerade erwähnten Mitteln ergibt diese Kombination den Satz: „To be or not to be that is the question“. Es steht Ihrer Kreativität nichts im Wege – entdecken Sie wieder Ihr Kind in sich selbst und basteln Sie sich so einige starke Passwörter, die Ihre Accounts gut beschützen können.

#### *Achtung*

Bitte achten Sie immer darauf, über welche Leitung (sicher/unsicher) Sie diese Passwörter letztendlich übertragen. Erinnern Sie sich an mail, pop, ftp, telnet, alle diese Programme übertragen den gesamten Netzwerkverkehr im Klartext. So kann ein böser Mensch durch einen Netzwerksniffer, wie wir ihn mit tcpdump kennengelernt haben, Ihre Userdaten herausfiltern.

Die Art, wie Applikationen mit der Anmeldung am System umgehen, kann man mit **PAM** (Pluggable Authentication Modules) einstellen. Mit einem PAM-Modul kann man beispielsweise die

Authentifizierung eines Users am System von der Abfrage eines **LDAP** (Light Weight Directory Protocol)-Server abhängig machen.

Die Pam-Module sind dynamisch ladbare Librarys, so, dass die eigentliche Applikation nicht neuerlich kompiliert werden muss. Oder man kann mit PAM-Modulen erreichen, dass die Authentifizierung am System über eine biometrische Komponente, wie Fingerabdruckscanner, erfolgen soll. Nicht nur die Anmeldung am System selbst kann mit PAM erfolgen, sondern auch andere Applikationen, wie z. B. der Apache Web-Server, können zur Authentifizierung der User, PAM verwenden. Um zu erkennen, ob eine Applikation PAM-Module unterstützt, führen wir folgenden Befehl aus.

```
#ldd passwd
libldap.so.2 => /usr/lib/libldap.so.2 (0x40031000)
  liblber.so.2 => /usr/lib/liblber.so.2 (0x40064000)
  libxcrypt.so.1 => /lib/libxcrypt.so.1 (0x40070000)
  libnsl.so.1 => /lib/libnsl.so.1 (0x400a4000)
libpam_misc.so.0      =>      /lib/libpam_misc.so.0
(0x400ba000)
libpam.so.0 => /lib/libpam.so.0 (0x400bd000)
  libdl.so.2 => /lib/libdl.so.2 (0x400c5000)
. . .
```

Die für uns interessanten Teile sind die beiden **libpam**-Libraries, die uns mitteilen, dass das Programm passwd mit PAM-Modulen umgehen kann. Diese Module sind im Verzeichnis **/etc/pam.d/\*** ihren Eigenschaften entsprechend konfigurierbar. Die Einträge sind wie folgt:

```
module-type    control-flag    module-path    argumente
```

So finden wir die für das passwd-Programm zuständige Datei in **/etc/pam.d/passwd** mit folgendem Inhalt:

```
##PAM-1.0
auth required pam_unix2.so nullok
account required pam_unix2.so
password required pam_pwcheck.sonullok
password required pam_unix2.so nullok use_first_pass    use_authtok
#password required pam_make.so /var/yp
session required pam_unix2.so
```

Neben der Verwendung von PAM-Modulen lässt sich die Sicherheit der Rechner mit Beschränkungen für die User erhöhen. Warum sollte man die User einschränken? Eine der Möglichkeiten, einem System bzw. dem Unternehmen Schaden zuzufügen, sind sogenannte **DoS** (Denial of Service)-Attacken. Diese sind dadurch gekennzeichnet, dass ein User oder ein Prozess eines Users so viel an System-Ressourcen verbraucht, dass keine mehr übrig bleiben, um den eigentlichen Dienst zu erbringen. Mit der Einschränkung der User beim Verbrauch der System-Ressourcen, kann man einige dieser DoS-Angriffe bereits abwehren. Die so definierten und gewünschten Einschränkungen können wir in der Datei **/etc/security/limits.conf** vornehmen. Die Einträge in dieser Datei haben folgendes Format:

```
domain type item wert
```

Unter domain kann entweder ein Username oder ein Gruppenname (@gruppe) oder ein \* für alle definiert werden. Beim Typ kann man unter hard oder soft wählen. Das item gibt an, was limitiert werden soll und der Wert – nun ja, den limitierenden Wert.

**Tabelle 187: Items, die limitiert werden können**

<b>Limit</b>	<b>Bedeutung</b>
core	Limitiert das core-File in K.
fsize	Limitiert die maximale Dateigröße in K.
data	Limitiert die maximale Daten-größe in K.
nofile	Limitiert die maximale Anzahl offener Dateien.
stack	Limitiert die maximale Stackgröße in K.
cpu	Limitiert die maximale CPU-Zeit für den User in Minuten.

nproc	Definiert die maximale Anzahl von Prozessen.
maxlogins	Beschränkt die maximale Anzahl der Logins für einen User.

Man kann z. B. für eine Gruppe ein soft-limit und ein hard-limit setzen. Das sieht dann so aus:

```
@lager      soft nfile      20
@lager      hard nfile      50
```

Der Unterschied liegt darin, dass das soft-limit überschritten werden darf bis zum hard-limit. Dieses hard-limit kann, wenn es einmal gesetzt wurde, nicht mehr überschritten werden.

Die Benutzer können auch mit dem Befehl **ulimit** beschränkt werden.

```
#ulimit -n 50
```

Damit hätten wir wieder die mögliche Anzahl gleichzeitig offener Dateien des Users auf 50 limitiert. Dieses Limitieren kann man z. B. in den Ressourcdateien der Shell unterbringen.

*Frage:*

Welche Ressourcdateien verwendet die bash? Worin unterscheiden sich diese Dateien?

**Tabelle 188: Optionen für den ulimit-Befehl**

<b>Option</b>	<b>Bedeutung</b>
-a	Zeigt alle derzeitig gesetzten Limits an.
-c WERT	Limitiert das Maximum der core-Datei.
-s WERT	Setzt die Maximale Stack-Größe.
-t WERT	Setzt die CPU-Zeit in Sekunden.
-p WERT	Limitiert die Pipe-Größe.
-n WERT	Setzt die maximale Anzahl der offenen Dateien.

-u WERT	Legt die maximale Anzahl der Prozesse fest, die der User starten kann.
-v WERT	Beschränkt die Menge des virtuellen Speichers, die ein User verbrauchen darf.

In den Bereich der Beschränkungen fallen auch die Themenbereiche der Beschränkung des verwendbaren Festplattenplatzes<sup>9</sup>.

Eine weitere Sicherheitslücke sind Standardinstallationen. Eine Standardinstallation ist eine Installation nach einem – von irgendjemanden definierten – Standard. Dieser Standard muss bei Ihnen aber gar nicht zutreffend sein, denn Ihre Firma ist nicht Standard, sondern etwas ganz Besonderes.

Bei Standardinstallationen gibt es immer wieder die sogenannten Standardpasswörter und die Standarddienste.

Die Standardpasswörter gehören sofort geändert, denn nicht nur Sie wissen ob deren Existenz und wie sie lauten, sondern auch die bösen Buben. Bei jeder Software, die eine eigene Benutzerverwaltung besitzt, gibt es mindestens einen administrativen Account. Er hat dann entweder keines (was wohl das Allerschlimmste ist, das einem passieren kann) oder irgendein Standardpasswort.

Standarddienste werden mitinstalliert. So könnte z. B. der Web-Server oder der Mail-Server oder der . . . xy-Server bei der Installation des Rechners gleich mitinstalliert und automatisch gestartet werden, ohne dass Sie es erfahren würden. Aber jeder Dienst, den ein Rechner auf einem Netzwerk anbietet, ist ein möglicher Angriffspunkt, um den Rechner zu kompromittieren. Sie sollten tunlichst überprüfen, ob alle Dienste, die auf Ihrem System laufen, auch die von Ihnen gewünschten Dienste erbringen. Alle anderen Services sollten Sie umgehend abschalten.

### Übung

Stellen Sie fest, welche Dienste auf Ihrem System laufen. Versuchen Sie einen oder mehrere dieser Dienste zu deaktivieren.

Wie sie dies machen? Einfach mit dem Befehl `netstat` die Dienste überprüfen und die nicht gewünschten Services in den entsprechenden Runlevels deaktivieren, indem Sie den Link auf die entsprechende Datei im Verzeichnis `/etc/init.d` löschen. Ach-

---

<sup>9</sup> Vergleiche Kapitel 4.3

ten Sie aber darauf, dass der Internet Super Daemon `inetd` oder `xinetd` nur Dienste starten kann, die konfiguriert werden.

Es gibt z. B. eine Gruppe von Diensten - die sogenannten **r-Services**. Das **r** steht hier für **remote**. Es gibt **rlogin** (remote Login), **rsh** (remote Shell) und **rcp** (remote copy). Diese Dienste werden über den Internet Super Daemon gestartet und mittels den Dateien `/etc/bosts.equiv` und `.rhosts` konfiguriert. Nachdem diese Dienste aber so gefährlich sind, möchte ich hier gar nicht mehr näher darauf eingehen, sondern möchte Sie bitten, diese Funktionalitäten über die Secure-Shell bzw. Secure-Copy zu realisieren. Achten Sie darauf, dass Ihr Internet Super Daemon so konfiguriert ist, dass diese Dienste nicht gestartet werden können.

## 16.2

### Walle, Walle - Firewall

Ich möchte hier keine Abhandlung über das Thema Firewall durchführen, denn dafür gibt es wieder eigene Bücher. Im Folgenden sei allerdings dargestellt, was wir mit den Bordmitteln, die uns Linux bereits mitliefert, durchführen können.

Was wollen wir erreichen? Wir wollen erreichen, dass wir ungewollte Verbindungen zu unserem Rechner unterbinden können, also nicht zulassen wollen. Wir wollen wieder folgende Logik als Prämisse verwenden: „Alles, was nicht ausdrücklich erlaubt ist, soll verboten sein“.

Lehnen Sie sich aber jetzt nicht zurück und sagen sich, damit brauche ich bei den Standarddingen nicht mehr aufpassen, wenn ich ohnehin keine Verbindungen dorthin zulasse. Erst in Verbindung mit allen Komponenten wird das System richtig rund. Es ist so wie in einer guten Küche. Der Braten wird erst dann so richtig schmecken, wenn alle Gewürze in der richtigen Menge und der richtigen Reihenfolge appliziert worden sind.

Das Ziel, nur Verbindungen zuzulassen, die unser Vertrauen besitzen, wird auf Paketebene durchgeführt. Wir analysieren also zu diesem Zwecke jedes Paket, das in unseren Rechner über die Schnittstellenkarte herein und heraus will. Diese Art von Kontrolle oder Filterung der Pakete nennt man **Packetfilter Firewall**.

Unter Linux stehen uns dazu zwei Systeme zur Verfügung. Bei Kernel-Versionen vor 2.4 heißt dieses System **ipchains** und



nach 2.4 **iptables**. Beginnen wir mit dem ipchains-System. Wie es der Name schon zum Ausdruck bringt, handelt es sich dabei um eine Verknüpfung von Ketten, die ineinander greifen. Als Grundvoraussetzung dafür ist wiederum die Unterstützung des Kernels notwendig. Wenn also Ihr Kernel wider Erwarten diese Paketfilterfunktionalität noch nicht integriert hat, bauen Sie sich bitte einen neuen. Alle modernen Distributionen haben diese Unterstützung bereits defaultmäßig aktiviert.

Wenn diese Unterstützung aktiviert ist, stehen uns zu Filterzwecken drei Ketten zur Verfügung. Diese heißen **input** (alles, das in die Interfaces reingeht), **output** (alles, was aus den Interfaces raus geht) und **forward** (alles, das zwischen zwei Interfaces transportiert werden soll).

Das ipchains-System ist sehr mächtig und zugleich auch sehr einfach zu bedienen, wenn man einmal das dahinter liegende Konzept durchschaut hat. Die Grundsyntax des ipchains-Systems bzw. des gleichlautenden Befehls ist:

ipchains Kommando Kette optionen

Damit wir kontrollieren können, ob irgendwelche Regeln bereits gesetzt wurden, tippen wir folgenden Befehl ein:

```
#ipchains -L
Chain INPUT (policy ACCEPT)
target      prot opt source                               destination

Chain FORWARD (policy ACCEPT)
target      prot opt source                               destination

Chain OUTPUT (policy ACCEPT)
target      prot opt source                               destination
```

Damit erkennen wir, dass auf unserem Rechner alles erlaubt ist, was herein oder heraus will.

**Tabelle 189: Kommandos für ipchains**

<b>Kommando</b>	<b>Bedeutung</b>
-A	Fügt eine Regel an das Ende der angegebenen Kette.
-D	Löscht eine Regel am Ende der angegebenen Kette.
-I NUM	Fügt eine Regel vor der mit NUM angegebenen Regel in der spezifizierten Kette ein.
-L	Listet alle Regeln auf.
-F	Löscht alle Regeln in der angegebenen Kette.
-Z	Setzt alle Zähler in der angegebenen Kette zurück.
-N Name	Erzeugt eine neue User-definierte Kette.
-P	Setzt die default-Regel der angegebenen Kette.

**Tabelle 190: Optionen für ipchains**

<b>Option</b>	<b>Bedeutung</b>
-p	Protokollangabe. Z. B. tcp oder udp.
-s	Source Adresse mit möglicher Portangabe.
-d	Destination-Adresse mit möglicher Portangabe.
--icmp-type	Kontrolliert den ICMP-Typ.
-j	Definiert das sogenannte Target, an das verzweigt werden soll.

-i	Definiert das Interface, für den diese Regel gelten soll.
-f	Diese Regel trifft nur für fragmentierte Pakete zu.
-b	Die gleiche Regel für beide Richtungen

Für das Target sind folgende Angaben möglich:

**Tabelle 191: Targets**

<b>Target</b>	<b>Bedeutung</b>
ACCEPT	Das passende Paket darf passieren.
DENY	Das passende Paket wird verworfen.
REJECT	Das entsprechende Paket wird zurückgewiesen.
REDIRECT	Das Paket wird umgeleitet.
MASQ	Damit wird eine Absende-Adresse in eine neue Adresse übersetzt - maskiert. Dies kann nur in der forward-Kette vorkommen

Wenn man seine Listen mit Regeln für alle Standard-Ketten und auch für die selbst definierten Ketten fertig hat, muss man noch einen kleinen Blick auf die Reihenfolge werfen. Denn die Listen werden von oben nach unten sequentiell abgearbeitet, und wenn die erste Regel gefunden wird, die auf das jeweilige Paket zutrifft, wird diese Regel ausgeführt und alle anderen nachfolgenden Regeln werden ignoriert. Obacht auf die Reihenfolge, insbesondere wenn man später Regeln hinzufügt und übersieht, dass für diese Art von Kontakt bereits eine Regel weiter oben in der Liste passt. Wenn keine Regel auf das Paket passt, gewinnt die Default-Regel, die immer auf DENY stehen sollte!

Die Forward-chain hat noch eine kleine Besonderheit. Diese chain ist in der Lage, die Absende-Adresse zu überschreiben und zwar in eine, die einem angeschlossenen Interface entspricht.

Damit können wir die Adressen eines privaten, internen Netzwerks von der inneren Netzwerkkarte über die Forward-chain in die offizielle Adresse des äußeren Netzwerkinterfaces übersetzen lassen und somit ein Internetgateway für ein Netzwerk realisieren. Das Target, das in der Forward-chain eingesetzt werden muss, ist **MASQ**. Wenn wir das Forwarding verwenden wollen, müssen wir dies dem Kernel auch noch mitteilen. Das erfolgt durch Setzen einer Eins in der Datei **ip\_forward**.

Wir setzen einfach folgenden Befehl ab:

```
#echo "1" > /proc/sys/net/ipv4/ip_forward
```

Damit haben wir das Forwarding aktiviert.

Alle diese Einstellungen sind allerdings nur von begrenzter Haltbarkeit. Nach einem Reboot ist alles wieder weg. Um das zu vermeiden, sollte man die Firewall-Regeln in einem Shell-Skript abspeichern und dieses durch ein Start-Skript im richtigen Runlevel ausführen lassen, dann ist alles O.K. Beim zweiten System – **iptables** – verhält sich alles analog. Bei **iptables** sind noch komplexere und durch **netfilter** noch feinere Einstellungen für die Kontrolle möglich.

Betrachten wir ein etwas umfangreicheres Beispiel. Die abgebildete Struktur ergibt eine Firewallarchitektur mit einer **DMZ** (Demilitarisierte Zone). Das interne Netzwerk ist in ein Schulungsnetzwerk und in ein Verwaltungsnetzwerk unterteilt.

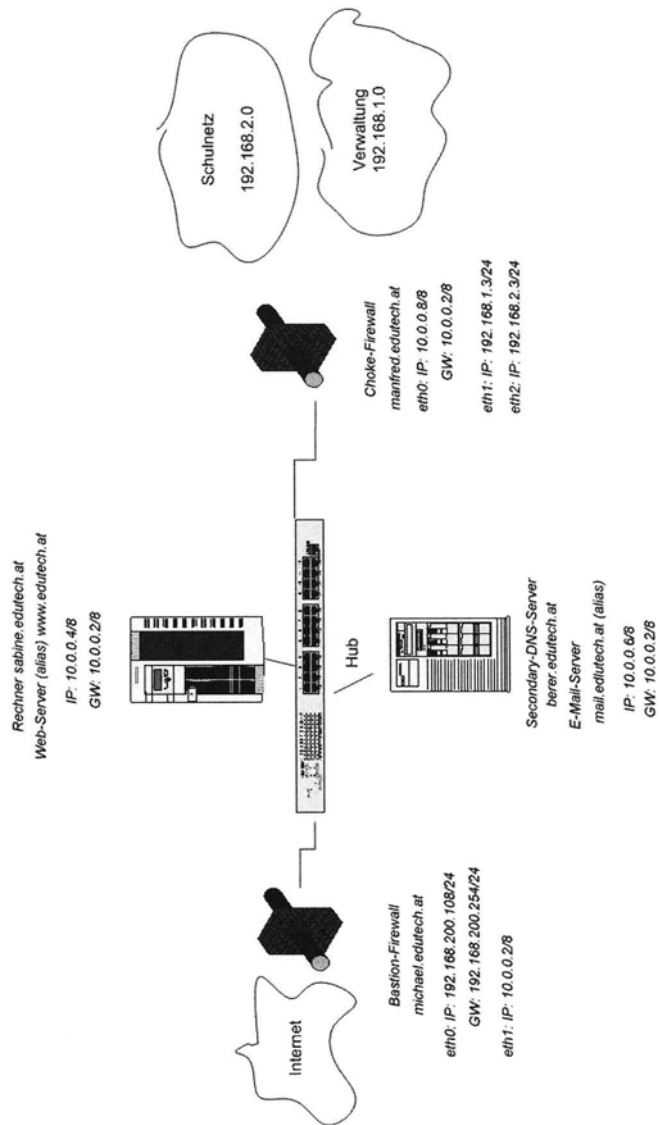


Abbildung 36: Firewallarchitektur mit DMZ

Beide Netzwerke haben einen Internetzugang über die innere Firewall mittels eines Proxy-Servers. Da in diesem Projekt ein Produktionsserver im internen Netzwerk abgefragt werden sollte, musste auch ein sogenannter Reversed-Proxy berücksichtigt werden. Dies ist nichts anderes als die normale Proxy-Funktion-

nalität, in umgekehrter Richtung, also vom Internet in das Intranet.

Im Folgenden ist ein mögliches Firewall-Skript aufgelistet. Dieses Firewall-Skript ist während einer Abschluss-Projektarbeit am Salzburg Research, welche ich als Trainer begleiten durfte, entstanden.

```
#!/bin/sh
#
# Startup script for the firewall configuration

#Hier werden die Variablen definiert

#Hier werden statt INTERFACES Variablen gesetzt.
vi="eth1"
si="eth2"
di="eth0"

#Hier werden statt NETZE Variablen gesetzt.
pn="192.168.200.0/24"
vn="192.168.1.0/24"
sn="192.168.0.0/24"
dn="10.0.0.0/8"
all="0.0.0.0/0"

#Hier werden statt HOSTS Variablen gesetzt.
ifw="10.0.0.8"
efw="10.0.0.2"
www="10.0.0.4"
mail="10.0.0.6"
pdns=$www
sdns=$mail
ifws="192.168.0.3"
ifwv="192.168.1.3"

#####
####

case "$1" in
    start)
        echo -n "Starting firewall: "
        echo
        #####
        ####

#Firewall flush. Alle firewallregeln werden gelöscht.
ipchains -F
```

```

#Forwarding wird aktiviert
echo 1 > /proc/sys/net/ipv4/ip_forward

#Es wird jeglicher Datentransfer per Default verboten.
ipchains -P input DENY
ipchains -P forward ACCEPT
ipchains -P output DENY

#Masquerading wird aktiviert.
#Alle Anfragen vom $sn Netz werden in Absender $ifw maskiert.
#Alle Anfragen vom $vn Netz werden in Absender $ifw maskiert.

ipchains -A forward -s $sn -d $dn -j MASQ
ipchains -A forward -s $vn -d $dn -j MASQ
ipchains -A forward -s $dn -d $vn -j MASQ

#Nun wird das Loopback-Interface freigegeben da dies fuer interne Zwe-
cke verwendet wird.
ipchains -A input -i lo -j ACCEPT
ipchains -A output -i lo -j ACCEPT

##### R E G E L N      A B      H I E R #####

##### DOMAIN NAME SERVER
#Freigabe: Primary DNS fuer Schulnetz
ipchains -A input -i $si -p UDP -s $sn 1024: -d $pdns 53 -j ACCEPT
ipchains -A output -i $di -p UDP -s $ifw 1024: -d $pdns 53 -j ACCEPT
ipchains -A input -i $di -p UDP -s $pdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $si -p UDP -s $pdns 53 -d $sn 1024: -j ACCEPT

ipchains -A input -i $si -p TCP -s $sn 1024: -d $pdns 53 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $pdns 53 -j ACCEPT
ipchains -A input -i $di -p TCP -s $pdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $si -p TCP -s $pdns 53 -d $sn 1024: -j ACCEPT

#Freigabe: Secondary DNS fuer Schulnetz
ipchains -A input -i $si -p UDP -s $sn 1024: -d $sdns 53 -j ACCEPT
ipchains -A output -i $di -p UDP -s $ifw 1024: -d $sdns 53 -j ACCEPT
ipchains -A input -i $di -p UDP -s $sdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $si -p UDP -s $sdns 53 -d $sn 1024: -j ACCEPT

ipchains -A input -i $si -p TCP -s $sn 1024: -d $sdns 53 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $sdns 53 -j ACCEPT
ipchains -A input -i $di -p TCP -s $pdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $si -p TCP -s $pdns 53 -d $sn 1024: -j ACCEPT

#Freigabe: Primary DNS für Verwaltungsnetz
ipchains -A input -i $vi -p UDP -s $vn 1024: -d $pdns 53 -j ACCEPT

```

```
ipchains -A output -i $di -p UDP -s $ifw 1024: -d $pdns 53 -j ACCEPT
ipchains -A input -i $di -p UDP -s $pdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p UDP -s $pdns 53 -d $vn 1024: -j ACCEPT
```

```
ipchains -A input -i $vi -p TCP -s $vn 1024: -d $pdns 53 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $pdns 53 -j ACCEPT
ipchains -A input -i $di -p TCP -s $pdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p TCP -s $pdns 53 -d $vn 1024: -j ACCEPT
```

#Freigabe: Secondary DNS für Verwaltungsnetz

```
ipchains -A input -i $vi -p UDP -s $vn 1024: -d $sdns 53 -j ACCEPT
ipchains -A output -i $di -p UDP -s $ifw 1024: -d $sdns 53 -j ACCEPT
ipchains -A input -i $di -p UDP -s $sdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p UDP -s $sdns 53 -d $vn 1024: -j ACCEPT
```

```
ipchains -A input -i $vi -p TCP -s $vn 1024: -d $sdns 53 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $sdns 53 -j ACCEPT
ipchains -A input -i $di -p TCP -s $pdns 53 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p TCP -s $pdns 53 -d $vn 1024: -j ACCEPT
```

##### WEB-SERVER

###Freigabe des www Web-Server in der Dmz für das Schulnetz

```
#ipchains -A input -i $si -p TCP -s $sn 1024: -d $www 80 -j ACCEPT
#ipchains -A output -i $di -p TCP -s $ifw 1024: -d $www 80 -j ACCEPT
#ipchains -A input -i $di -p TCP -s $www 80 -d $ifw 1024: -j ACCEPT
#ipchains -A output -i $si -p TCP -s $www 80 -d $sn 1024: -j ACCEPT
```

##### PROXY

##Freigabe des www Servers ueber Proxy in der Dmz für das DMZ Netz

```
#ipchains -A input -i $di -p TCP -s $dn 1024: -d $ifw 8080 -j ACCEPT
#ipchains -A output -i $di -p TCP -s $ifw 1024: -d $www 80 -j ACCEPT
#ipchains -A input -i $di -p TCP -s $www 80 -d $ifw 1024: -j ACCEPT
#ipchains -A output -i $di -p TCP -s $ifw 8080 -d $dn 1024: -j ACCEPT
```

##Freigabe des www Servers ueber Proxy in der Dmz für das SCHULNETZ

```
#ipchains -A input -i $si -p TCP -s $sn 1024: -d $ifw 8080 -j ACCEPT
#ipchains -A output -i $di -p TCP -s $ifw 1024: -d $www 80 -j ACCEPT
#ipchains -A input -i $di -p TCP -s $www 80 -d $ifw 1024: -j ACCEPT
#ipchains -A output -i $si -p TCP -s $ifw 8080 -d $sn 1024: -j ACCEPT
```

##### TRANSPARENTER PROXY

##Freigabe des www Servers ueber T-Proxy in der Dmz für das SCHULNETZ

```
ipchains -A input -i $si -p TCP -s $sn 1024: -d $all 80 -j REDIRECT 8080
ipchains -A input -i $si -p TCP -s $sn 1024: -d $all 8080 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $all 80 -j ACCEPT
ipchains -A input -i $di -p TCP -s $all 80 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $si -p TCP -s $all 80 -d $sn 1024: -j ACCEPT
```



```

##Freigabe des www Servers ueber T-Proxy in der Dmz für das VERWAL-
TUNGSNETZ
ipchains -A input -i $vi -p TCP -s $vn 1024: -d $all 80 -j REDIRECT
8080
ipchains -A input -i $vi -p TCP -s $vn 1024: -d $all 8080 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $all 80 -j ACCEPT
ipchains -A input -i $di -p TCP -s $all 80 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p TCP -s $all 80 -d $vn 1024: -j ACCEPT

##### REVERST PROXY
###Freigabe des www Web-Server in dem Verwaltungsnetz über dmz reverst
#proxy
ipchains -A input -i $di -p TCP -s 10.0.0.10 1024: -d 192.168.1.2 80 -j AC-
CEPT
ipchains -A output -i $vi -p TCP -s $ifwv 1024: -d 192.168.1.2 80 -j ACCEPT
ipchains -A input -i $vi -p TCP -s 192.168.1.2 80 -d $ifwv 1024: -j ACCEPT
ipchains -A output -i $di -p TCP -s 192.168.1.2 80 -d 10.0.0.10 1024: -j AC-
CEPT

##### MAIL
#Freigabe des Mail-Server in der Dmz für das Verwaltungsnetz
ipchains -A input -i $vi -p TCP -s $vn 1024: -d $mail 25 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $mail 25 -j ACCEPT
ipchains -A input -i $di -p TCP -s $mail 25 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p TCP -s $mail 25 -d $vn 1024: -j ACCEPT

ipchains -A input -i $vi -p TCP -s $vn 1024: -d $mail 110 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 1024: -d $mail 110 -j ACCEPT
ipchains -A input -i $di -p TCP -s $mail 110 -d $ifw 1024: -j ACCEPT
ipchains -A output -i $vi -p TCP -s $mail 110 -d $vn 1024: -j ACCEPT

##### SSH
#Freigabe der SSH fürs DMZ Netz
ipchains -A input -i $di -p TCP -s $www 1024: -d $ifw 22 -j ACCEPT
ipchains -A output -i $di -p TCP -s $ifw 22: -d $www 1024: -j ACCEPT

##### ICMP Pakete erlauben
#ipchains -A input -i $di -p ICMP -s $all -d $ifw -j ACCEPT
#ipchains -A output -i $di -p ICMP -s $ifw -d $all -j ACCEPT

ipchains -A input -p ICMP -j ACCEPT
ipchains -A output -p ICMP -j ACCEPT
;;

stop)
    echo -n "Shutting down firewall: "
ipchains -F
ipchains -P input ACCEPT
ipchains -P forward ACCEPT
ipchains -P output ACCEPT

```

```
ipchains -A forward -s $sn -d $dn -j MASQ
ipchains -A forward -s $vn -d $dn -j MASQ
ipchains -A forward -s $dn -d $vn -j MASQ
    echo
    ;;
restart)
    $0 stop
    $0 start
    ;;
*)
    echo "Usage: $0 {start|stop|restart}"
    exit 1
esac

exit 0
```

Wenn wir mit dieser Firewall zufrieden sind, wollen wir zum Schluß auch ein Backup davon machen. Dazu können wir den Befehl **ipchains-save** verwenden.

```
# ipchains-save > my_firewall
Saving `input'.
Saving `output'.
Saving `forward'.
#
```

Das Einspielen des Backups erfolgt wiederum mit **ipchains-restore**.

```
# ipchains-restore < my_firewall
Restoring `input'.
Restoring `output'.
Restoring `forward'.
```

## 16.3

### Die Secure-Shell oder alles ist verschlüsselt

Nachdem ich Sie gebeten habe, die r-Kommandos und die Klar-text-Tools wie telnet nicht zu verwenden, sondern durch die Secure-Shell zu ersetzen, soll diese hier erklärt werden. Der Schutz, den die Secure-Shell bietet, ist gegeben durch die hohe Verschlüsselung. Bereits der Austausch der Userdaten erfolgt über einen verschlüsselten Kanal. Die Sicherheit kann aber noch ein Stück erhöht werden, indem man die **RSA**-Authentifikation aktiviert.

Die Secure-Shell liegt derzeit in zwei Versionen vor, die Version 1 und die Version 2. Der Unterschied zwischen den beiden Versionen liegt hauptsächlich in den Möglichkeiten der Authentifikationsmethoden.

Jeder Rechner besitzt einen Hostspezifischen RSA-Schlüssel mit der Länge von 1024 Bit. Wenn nun ein Client mit dem Server über **Port 22** Kontakt aufnimmt, dann reagiert der Server-Daemon mit seinem öffentlichen Schlüssel (auch Host-Schlüssel genannt).

Der Client wiederum vergleicht den Host-Schlüssel mit seiner Datenbank, um sicher zu sein, dass er sich nicht geändert hat. Der Client erzeugt eine 256 Bit lange zufällige Nummer und verschlüsselt diese unter Verwendung des Host-Schlüssels und des Server-Schlüssels. Diese verschlüsselte Nummer sendet der Client zum Server. Beide Seiten verwenden diese verschlüsselte Nummer als Session-Key, um die weitere Kommunikation zu verschlüsseln. Die restliche Kommunikation erfolgt verschlüsselt, wobei standardmäßig der 3DES-Algorithmus verwendet wird. Server und Client treten nun in einen Authentifikationsdialog. Der Client versucht, sich selbst mittels der **.rhosts** Identifikation oder **.rhosts** kombiniert mit RSA oder mit der RSA-Challenge-Response Authentifikation oder schließlich mittels Passwort zu identifizieren.

Die **.rhosts**-Authentifikation ist per Default deaktiviert, da sie sehr unsicher ist. Die Version 2 arbeitet ähnlich, bietet aber noch eine **DSA**-Methode zur Authentifikation. Der Secure-Shell Daemon, heißt **sshd** und wird mittels der Konfigurationsdatei **/etc/ssh/sshd\_config** konfiguriert.

**Tabelle 192: Dateien die zu sshd gehören**

<i><b>Datei</b></i>	<i><b>Bedeutung</b></i>
sshd_config	Beinhaltet Konfigurationsangaben für den sshd-Daemon.
ssh_host_key ssh_host_dsa_key ssh_host_rsa_key	Diese Dateien enthalten jeweils den privaten Teil des Host-Schlüssels. Diese Dateien muss root-Rechte besitzen. Der Server startet nicht, wenn diese Dateien welt-lesbar sind.

ssh_host_key.pub ssh_host_dsa_key.pub ssh_host_rsa_key.pub	Diese Dateien halten den öffentlichen Teil des verwendeten Schlüssels und müssen weltlesbar sein.
~/.ssh/authorized_keys	Speichert die öffentlichen Schlüssel, die verwendet werden können, um sich als dieser User anmelden zu können.
ssh_known_hosts ~/.ssh/known_hosts	Diese Dateien werden überprüft, wenn die rhosts und RSA-Authentifikation kombiniert zum Einsatz kommen.
/etc/nologin	Wenn diese Datei existiert, dann verweigert der sshd-Server alle nicht root-Verbindungen und zeigt nur den Inhalt dieser Datei an.
~/.ssh/rc /etc/ssh/sshr	Diese Dateien werden nach dem Setzen der Umgebung ausgeführt und können dazu verwendet werden, notwendige Initialisierungen auszuführen, noch bevor der User seine Shell zugewiesen bekommt. Die eine Datei ist pro User, die andere global gültig.

Nun wollen wir unseren sshd-Server konfigurieren. Wir wenden uns der Datei **/etc/ssh/sshd\_conf** zu und betrachten die für uns jetzt wichtigsten Einträge.

```
Port 22
#Protocol 2,1
ListenAddress 0.0.0.0
#ListenAddress ::

# HostKey for protocol version 1
HostKey /etc/ssh/ssh_host_key
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
```

```
# Lifetime and size of ephemeral version 1 server key
#KeyRegenerationInterval 1h
#ServerKeyBits 768

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes

RSAAuthentication yes
#PubkeyAuthentication yes
AuthorizedKeysFile .ssh/authorized_keys

# For this to work you will also need host keys in
#/etc/ssh/ssh_known_hosts
RhostsRSAAuthentication no
# similar for protocol version 2
HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# RhostsRSAAuthentication and HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no

# Change to no to disable s/key passwords
#ChallengeResponseAuthentication yes
```

Wir können den Server mit `/etc/init.d/sshd` start schon starten. Wenn wir wieder wünschen, dass der Secure-Shell-Server nach jedem Reboot gleich wieder gestartet werden soll, müssen wir das Start-Skript in den entsprechenden Runlevel verlinken.

Um das alles auszuprobieren, verwenden wir den Secure-Shell-Client **ssh** und geben in einem Konsolenfenster Folgendes ein:

```
#ssh helmut@tux
```

Hierbei ist `helmut` der Username und `tux` der Rechner, mit dem wir uns verbinden wollen.

Wenn wir dies das erste Mal tun, sendet der Server seinen öffentlichen Schlüssel. Da wir ihn noch nicht in unserer Datenbank

haben, warnt uns das System, dass die Authentizität des Servers nicht geprüft werden kann und nicht gesichert ist. Des Weiteren werden Sie gefragt, ob Sie diesem Schlüssel vertrauen. Wenn Sie diese Frage mit ja beantworten, wird dieser Schlüssel aufgenommen, und beim nächsten Mal wird dieser Warnhinweis nicht mehr erscheinen, erst dann, wenn sich der öffentliche Schlüssel des Servers geändert hat.

Danach werden Sie nach Ihrem Passwort gefragt. Wenn Sie hier Ihr richtiges Passwort eingeben, erhalten Sie eine Shell, auf der Sie arbeiten können, als ob Sie direkt vor dem betreffenden Rechner sitzen würden.

Den Abschluss soll eine Tabelle mit den wichtigsten Optionen für den Secure-Shell-Client machen.

**Tabelle 193: Optionen für den ssh-Client**

<b><i>Option</i></b>	<b><i>Bedeutung</i></b>
-c WERT	Angabe des zu verwendenden Verschlüsselungsalgorithmus. (3DES, DES, ...).
-c LISTE	Bei der Version 2 des ssh-Protokolls kann man eine Liste von Algorithmen angeben und zwar in der gewünschten Reihenfolge.
-e WERT	Setzt den Escape-Character.
-g	Gestattet es remote-Rechnern, sich an lokal umgeleitete Ports zu wenden.
-i DATEI	Verwendet DATEI als Identifikationsdatei.
-I DEV	Definiert, welches SmartCard-Gerät benutzt werden soll.
-l USER	Definiert den User, der sich anmelden will.

-p PORT	Angabe des Ports, mit dem man sich verbinden will, falls nicht der Standardport 22 verwendet wird.
-l oder -2	Angabe der Portokollversion.

Um anstelle von Username und Passwort die RSA-Authentifizierung zu verwenden, muss der jeweilige Benutzer zuerst ein Schlüsselpaar generieren. Dies kann er mit dem Tool **ssh-keygen** tun.

```
#ssh-keygen -t rsa
Generating public/private rsa key pair.
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
44:43:c0:b8:4c:3d:cb:11:2f:d8:6b:02:c1:ab:44:31 root@servus
```

**Tabelle 194: Optionen für ssh-keygen**

<b>Optionen</b>	<b>Bedeutung</b>
-b BIT	Definiert die Länge der Schlüssel, die erzeugt werden sollen.
-f DATEI	Definiert den Dateinamen für den Schlüssel.
-l	Zeigt den Fingerprint einer spezifizierten Datei an.
-p	Ändert die Passphrase.
-y	Liest einen Privat Key ein und gibt einen Public Key aus.
-t TYP	Definiert den Schlüsseltyp, der erzeugt wird.
-D READER	Lädt den Public Key aus dem angegebenen SmartCard-Lesegerät.
-U READER	Lädt den Privat Key in den SmartCardLeser READER hoch.

**OBJECTIVE:** 1.101.1 **TYPE:** mcma

Wo können bei Linux-Systemen Probleme auftreten, wenn die BIOS Einstellung "LBA-Modus" nicht aktiviert wird?

1. Linux umgeht das BIOS bei Festplattenzugriffen, es können keine Probleme auftreten
2. bei Partitionierungssoftware
3. beim Linux Dateisystem "ext2", das den LBA-Modus benötigt
4. bei Bootmanagern, die unter Umständen ohne LBA-Modus für das BIOS nicht mehr erreichbar sind
5. LBA bedeutet Linux BootAble. Der LBA-Modus ist für Linux-Systeme zwingend erforderlich

**OBJECTIVE:** 1.101.3 **TYPE:** mc

Was versteht man unter dem Begriff "Winmodem"?

1. Modems, die über einen eigenen Befehlssatz über die serielle Schnittstelle angesprochen werden
2. Modems, die eigene Treiber benötigen, um angesprochen werden zu können
3. Modems, die nur über einen eigenen Microsoft-spezifischen Befehlssatz angesprochen werden können
4. Modems, die über die USB-Schnittstelle angesprochen werden

**OBJECTIVE:** 1.101.4 **TYPE:** mcma

Welche Aussagen treffen auf einen SCSI-Bus zu?



1. Der SCSI-Bus darf eine Gesamtlänge von 3 Metern nicht überschreiten
2. Da der Bus Hot-Plug-fähig ist, können freie Stecker offen gelassen werden
3. Der Leitungsabstand zwischen den Geräten darf maximal 10 cm betragen
4. Der Bus darf nur zwei Enden haben
5. Für neue SCSI-Geräte muss immer der letzte freie Anschluss verwendet werden

**OBJECTIVE:** 1.101.5 **TYPE:** mc

Mit welchem Befehl kann man Informationen über Karten am PCI-Bus in numerischer Form ausgeben

1. pnpdump
2. lspci
3. cat /usr/share/misc/pci.ids
4. lspci -n
5. cat /proc/pci

**OBJECTIVE:** 1.101.5 **TYPE:** mc

Mit welchem Befehl leiten Sie PnP-Hardwareinformationen in die Standard-PnP-Konfigurationsdatei um?

1. isapnp > /etc/pnpdump.conf
2. pnpdump | /etc/isapnp.conf
3. pnpdump | /usr/share/misc/isapnp.conf
4. pnpdump > /etc/isapnp.conf
5. isapnp # /usr/conf/isapnp.conf

**OBJECTIVE:** 1.101.5 **TYPE:** fitb

# Minimum IO Base address 0x280

---

# Maximum IO Base address 0x2a0

# IO base alignment 8 bytes

Welche IO-Basisadressen wären für diese Karte gültig?

**OBJECTIVE:** 1.101.6 **TYPE:** mc

Welches "setserial"-Flag steht für die doppelte Baudrate bei der Kommunikation zwischen Computer und Modem?

1. spd\_hi
2. spd\_warp
3. spd\_shi
4. spd\_vhi
5. spd\_normal

**OBJECTIVE:** 1.101.7 **TYPE:** mcma

Wie lautet der Befehl zum Aktivieren des USB-Geräte-Dateisystems?

1. mount -t usbdevfs none /proc/bus/usb
2. mount -t usbdevfs /dev/bus/usb /mnt/usb
3. modprobe usb
4. Befehl "none /proc/bus/usb usbdevfs defaults 0 0" in /etc/fstab
5. rcusbdevfs start

**OBJECTIVE:** 1.102.1 **TYPE:** mc

Wie groß würden Sie die Swap-Partition auf einem System mit 512MB-RAM Speicher sinnvoller Weise wählen?

1. 512 MB
2. 1024 MB
3. 256-320 MB
4. 64-128 MB
5. 8-16 MB

**OBJECTIVE: 1.102.1 TYPE: mcma**

Welche der folgenden Dateisysteme sollten auf einem Fileserver eigene Partitionen erhalten?

1. /var
2. /usr
3. /home
4. /tmp
5. /etc

**OBJECTIVE: 1.102.1 TYPE: mcma**

Wobei handelt es sich um Vorteile bei der Benutzung mehrerer Partitionen unter Linux?

1. Die Festplattenzugriffszeiten werden erheblich verkürzt
2. Userquotas (Einschränkungen) bestehen aus virtuellen Partitionen
3. Backups werden leichter realisierbar
4. Die 1024 Zylinder-Grenze wird dadurch wirkungslos
5. Die Systempartition kann vor Usern abgeschirmt und somit vor Speicherplatzmangel bewahrt werden

**OBJECTIVE: 1.102.1 TYPE: mc**

Welche Aufgabe hat das /var/lock-Verzeichnis?

1. Es dient zum Identifizieren von PIDs und ihren Prozessen
2. Es speichert Instanzen systemkritischer Prozesse
3. Es verhindert, dass System-Ressourcen mehrfach vergeben werden
4. Es verhindert, dass Programme mehrfach gestartet werden

---

**OBJECTIVE:** 1.102.1 **TYPE:** mc

Welche UNIX – Dateisysteme sollten read-only gemountet werden?

1. /var
2. /usr
3. /home
4. /tmp
5. /boot

**OBJECTIVE:** 1.102.2 **TYPE:** mc

Mit welchem Befehl kann LILO aus dem MBR entfernt werden?

1. lilo -d
2. lilo -u
3. mbr -n
4. lilo -mbr

**OBJECTIVE:** 1.102.3 **TYPE:** mcma

Wie lautet der Befehl zum Entpacken eines mit "bzip2" gepackten tar-Archivs?

1. bzcat *Dateiname*.tar.bz2 | tar -xvf-
2. bzcat *Dateiname*.tar.bz2
3. tar -xjvf *Dateiname*.tar.bz2
4. bzcat *Dateiname*.tar.bz2 | tar -xvf -
5. tar -xzvf *Dateiname*.tar.bz2

**OBJECTIVE:** 1.102.3 **TYPE:** mc

Wie kann ein Makefile erstellt werden?

1. configure
2. ./configure
3. ./configure (wenn das "configure"-Script existiert)
4. configure (wenn das "configure"-Script existiert)

**OBJECTIVE:** 1.102.3 **TYPE:** fitb

pre=/home/pat

dir=/appl

conf=\${pre}/\${dir}/etc

Welchen Wert enthält die Makefile-Variable "conf"?

**OBJECTIVE:** 1.102.3 **TYPE:** mc

Für welchen Befehl sind zumindest root-Rechte erforderlich?

1. ./configure
2. make all
3. make
4. make install
5. make clean

**OBJECTIVE:** 1.102.3 **TYPE:** fitb

- a) make install
- b) make clean
- c) Anpassen des Makefiles
- d) tar -xzf Archiv.tar.gz
- e) ./configure
- f) make

Sortieren Sie die Buchstaben in der richtigen Reihenfolge

---

**OBJECTIVE:** 1.102.3 **TYPE:** mcma

Kann auf die Installation von Archiven, in denen bereits ein Makefile existiert, noch Einfluss genommen werden?

1. Nein, die Makefiles werden automatisch aus dem Konfigurationsscript erstellt und liegen in binärer Form vor
2. Ja, im Makefile müssen in manchen Fällen noch Pfadanpassungen u. ä. vorgenommen werden
3. Indem man das "config"-Script nochmals ausführt, kann man ein altes Makefile mit neuen Einstellungen überschreiben
4. Man kann schon, jedoch sind spezielle Kenntnisse der "Makefile"-Scriptsprache erforderlich, weshalb davon abzuraten ist

**OBJECTIVE:** 1.102.4 **TYPE:** mc

Welche Umgebungsvariable enthält eine Liste der Library-Pfade?

1. LD\_LIBRARY\_PATH
2. LIBRARY\_PATH
3. LD\_LIB\_PATH
4. /etc/ld.so.cache

**OBJECTIVE:** 1.102.4 **TYPE:** mc

Welches Kommando gibt die vom Programm "vi" benutzten Shared Libraries aus?

1. ldd vi
2. ldconfig /bin/vi
3. ldd /bin/vi
4. ldd -l /bin/vi

**OBJECTIVE:** 1.102.5 **TYPE:** mc

Sie haben mehrere .deb-Pakete aus dem Internet zu Updatezwecken in einem Verzeichnis auf Ihrer Festplatte gespeichert. Mit welchem Kommando installieren Sie alle Pakete auf einmal?

1. `dpkg --install Updatepaket_x.deb` für alle Pakete in einem Script aufrufen
2. `dpkg --all -i /Verzeichnisname`
3. `dpkg [Optionen] -R /Verzeichnisname`
4. `dpkg [Optionen] --recursive Updatepaket_1.deb`

**OBJECTIVE:** 1.102.5 **TYPE:** mc

Mit welchem Kommando entfernen Sie ein Paket "webmin" vollständig aus dem System?

1. `dpkg -r webmin`
2. `apt-get --remove webmin`
3. `dpkg -P webmin_1.0.11_i386.deb`
4. `dpkg --purge webmin`
5. `remove webmin`

**OBJECTIVE:** 1.102.5 **TYPE:** mcma

Wie überprüfen Sie, ob ein Paket korrekt installiert wurde?

1. Mit `"dpkg -l | --list"` überprüfen, ob das Paket aufgelistet wird
2. Durch Ausführen der Anwendung
3. `apt-get --status Paketname`
4. `dpkg -p Paketname`
5. `dpkg -s Paketname`

**OBJECTIVE:** 1.102.5 **TYPE:** fitb

Wie lautet die dpkg-Option, mit der alle anderen Pakete mit deinstalliert werden, wenn ein von ihnen benötigtes Paket deinstalliert wird?

---

**OBJECTIVE:** 1.102.5 **TYPE:** mcma

Mit welchem Kommando kann ein bereits installiertes Paket neu konfiguriert werden?

1. `dpkg-reconfigure Paketname`
2. `dpkg -reconfigure Paketname`
3. Das ist nur möglich, wenn das Paket ein debconf-Skript enthält
4. Das ist überhaupt nicht möglich, man kann das Paket nur neu installieren

**OBJECTIVE:** 1.102.5 **TYPE:** mcma

Wie fügt man Installationsquellen für apt-get hinzu?

1. manuell in `/etc/apt/apt.conf.d`
2. manuell in `/etc/apt/sources.list`
3. `dselect => access => apt`
4. `dselect => select => apt`
5. `apt-get source Installationsquelle`

**OBJECTIVE:** 1.102.5 **TYPE:** mc

Sie wollen ein .pkg-Paket installieren, haben aber nur den Debian-Paketmanager zur Verfügung. Wie gehen Sie vor?

1. Das Paket ist hier unbrauchbar, ich suche einen entsprechenden tarball oder ein .deb-Paket aus dem Internet
2. `dpkg -from-pkg -i Paketname`
3. `alien Paketname.pkg; dpkg -i Paketname.deb`
4. `alien -to-deb Paketname.pkg; dpkg -i Paketname.deb`



**OBJECTIVE:** 1.102.6 **TYPE:** mc

Mit welchem Kommando deinstallieren Sie ein Paket?

1. `rpm -u Paketname`
2. `rpm -r Paketname`
3. `rm Paketname`
4. `rpm -e Paketname`

**OBJECTIVE:** 1.102.6 **TYPE:** mcma

Welchen rpm-Befehl würden Sie verwenden, um System-Pakete zu installieren?

1. `rpm -i -test -replacepkgs -replacefiles Paketname`
2. `rpm -i -excludedocs -oldpackage Paketname`
3. `rpm -i -replaceoldpackage -nodeps Paketname`
4. `rpm -i -replacepkgs -replacefiles Paketname`
5. `rpm -i -force Paketname`

**OBJECTIVE:** 1.102.6 **TYPE:** mcma

Mit welchen rpm-Befehlen können Updates durchgeführt werden?

1. `rpm -v Paketname`
2. `rpm -F [Optionen] Paketdatei`
3. `rpm [Optionen] -U Paketname`
4. `rpm -U [Optionen]`
5. `rpm -F Paketname [Optionen]`

**OBJECTIVE:** 1.102.6 **TYPE:** mc

Sie finden eine Datei, von der Sie nicht sicher sind, ob Sie von einem Paket benötigt wird. Wie überprüfen Sie die Zugehörigkeit?

1. `rpm -qc Dateiname`
2. `rpm -qf Dateiname`
3. `rpm -q --whatrequires Dateiname`
4. `rpm -V Dateiname`

---

**OBJECTIVE:** 1.102.6 **TYPE:** mc

Sie möchten überprüfen, welche Pakete für den Skriptinterpreter Perl installiert sind. Wie lautet der Befehl?

1. rpm -q "perl"
2. rpm -qa --group perl
3. rpm -qf /bin/perl
4. rpm -qa | grep perl

**OBJECTIVE:** 1.102.6 **TYPE:** mc

Sie haben ein Paket erhalten, bei dem Sie sich über die Quelle nicht ganz sicher sind. Wie können Sie das Paket überprüfen?

1. rpm -v *Paketname*
2. rpm -V *Paketname*
3. rpm -qv *Paketname*
4. rpm --test *Paketname*

**OBJECTIVE:** 1.102.6 **TYPE:** mc

Sie haben ein Paket mit rpm überprüft. Die Ausgabe lautet "..5.....". Welche Bedeutung hat diese Ausgabe?

1. Alle Tests bis auf den fünften wurden erfolgreich durchlaufen
2. Fünf Tests schlugen fehl
3. Die Zeitmarke der mtime stimmt nicht
4. Die Prüfsumme stimmt nicht

**OBJECTIVE:** 1.102.6 **TYPE:** fitb

Mit welchem Befehl kann die PGP/GPG - Signatur eines Paketes überprüft werden? (Nur der Befehl, ohne weitere Optionen)

**OBJECTIVE:** 1.103.1 **TYPE:** mc

Wie sind UNIX-Kommandos allgemein aufgebaut?

1. *Programm Dateipfad Optionen*
2. *Programm Optionen Dateipfad*
3. *Programm Dateipfad*
4. *Dateipfad Programm Optionen*
5. *Dateipfad | Programm Optionen*

**OBJECTIVE:** 1.103.1 **TYPE:** mc

Sie möchten zwei Programme gleichzeitig aufrufen, wobei *Programm1* im Vordergrund und *Programm2* im Hintergrund laufen soll. Wie lautet der Aufruf?

1. *Programm1 && Programm2*
2. *Programm2 & Programm1*
3. *Programm1 & Programm2*
4. *Programm2 ; Programm1*
5. *Programm1 ; Programm2*

**OBJECTIVE:** 1.103.1 **TYPE:** mc

Sie möchten die aktuelle Shell (bash) durch die C-Shell ersetzen. Mit welchem Befehl können Sie dies erreichen?

1. */bin/csh*
2. */bin/bash | /bin/csh*
3. *exec /bin/csh*
4. *exit; /bin/csh*

**OBJECTIVE:** 1.103.1 **TYPE:** mc

Was würde das folgende Kommando bewirken?

`vi `date +%Y%m%d%H%M`.log`

1. *Erstellt und öffnet die Datei ``date +%Y%m%d%H%M`.log` in vi*
2. *Erstellt und öffnet die Datei  `+%Y%m%d%H%M`.log` in vi*

- 
3. Erstellt ein leeres .log File mit dem momentanen Datum als Name und öffnet es in vi
  4. Erstellt ein leeres .log File mit dem momentanen Datum als Name
  5. Konfiguration für vi, die bewirkt, dass vi Logfiles führt, wann es von wem benutzt wurde

**OBJECTIVE:** 1.103.1 **TYPE:** mcma

Mit welchen Kommandos erhalten Sie nähere Informationen zu fast allen Befehlen?

1. *Befehl* -h
2. *Befehl* -?
3. *Befehl* --help
4. *Befehl* /?
5. man *Befehl*

**OBJECTIVE:** 1.103.2 **TYPE:** mcma

Sie möchten in einer Datei alle Groß- durch Kleinbuchstaben ersetzen. Mit welchen Befehlen wird dies ermöglicht?

1. tr
2. uniq
3. sed
4. paste
5. nl

**OBJECTIVE:** 1.103.2 **TYPE:** mc

Sie wollen die fünfte bis zehnte Zeile der Datei test.txt ausgeben. Wie lautet der Befehl?

1. vi --export l5-10 test.txt
2. cut -l5-10 test.txt
3. less -l4-9 test.txt
4. cat test.txt | tail -4l | head -10l
5. cat test.txt | head -10l | tail -4l

**OBJECTIVE:** 1.103.2 **TYPE:** mc

Sie möchten das Logfile eines Webservices laufend in einer Konsole anzeigen und so alle Änderungen in Echtzeit mitverfolgen. Wie lautet der Befehl?

1. `cat -R webservice.log`
2. `tail webservice.log`
3. `tail -f webservice.log`
4. `nl \: webservice.log`

**OBJECTIVE:** 1.103.2 **TYPE:** fitb

Wie lautet der Befehl, um große Dateien auf mehrere kleine mit bestimmter Bytegröße aufzuteilen? (Kommando und Optionen)

**OBJECTIVE:** 1.103.2 **TYPE:** mc

Sie wollen eine Liste aller User (die 5te Spalte der Datei `/etc/passwd`) des aktuellen Rechners ausgeben. Wie lautet der Befehl?

1. `split -l -d: -f5 /etc/passwd`
2. `sort -c5 /etc/passwd | cat`
3. `cut -d: -f5 /etc/passwd`
4. `cat -c5-13 -l /etc/passwd`

**OBJECTIVE:** 1.103.2 **TYPE:** mcma

Wie lautet der Befehl, um die Zeilenanzahl der Datei `test.txt` auszugeben?

1. `wc --lines test.txt`
2. `wc -c test.txt`
3. `wc --words test.txt`
4. `wc -l test.txt`

---

**OBJECTIVE:** 1.103.3 **TYPE:** mc

Wie lautet der Befehl, um alle Dateien des gegenwärtigen Verzeichnisses mit den wichtigsten Zusatzinformationen anzuzeigen, die mit einem Großbuchstaben beginnen und kein "b", "c" oder "d" als letztes Zeichen aufweisen?

1. `ls [A-Z]*[bcd]`
2. `dir [A-Z]*[!bcd]`
3. `ls -l [#]*[!bcd]`
4. `ls -l [A-Z]*[!bcd]`

**OBJECTIVE:** 1.103.3 **TYPE:** mc

Sie möchten alle Dateien mit der Endung .conf oder .misc aus /etc nach /root kopieren. Wie lautet der Befehl?

1. `mv /etc/*.[cm][oi][ns][fc] /root`
2. `cp /etc/*.[conf|misc] /root`
3. `cp /etc/*.[cm][oi][ns][fc] /root`
4. `cp /etc/*.[conf][misc] /root`

**OBJECTIVE:** 1.103.4 **TYPE:** mc

Sie möchten die Ausgaben eines Programms nach jedem Aufruf in eine Log-Datei schreiben, wobei die alten Einträge nicht überschrieben werden sollen. Wie lautet der Aufruf?

1. `Programm > Logdatei`
2. `Programm | Logdatei`
3. `Programm >> Logdatei`
4. `Programm << Logdatei`
5. `Programm > Logdatei 2>&1`

**OBJECTIVE:** 1.103.4 **TYPE:** mc

Welche kryptische Anweisung bündelt STDOUT und STDERR bei Angaben von Umleitungen?

1. 2&1
2. 2>1
3. 2>&1
4. 2&>1

**OBJECTIVE:** 1.103.4 **TYPE:** mc

Was ist der Vorteil vom Programm xargs gegenüber normaler Kommandosubstitution?

1. xargs kann beliebig viele Kommandos mit den selben Parametern aufrufen
2. xargs arbeitet schneller als die Kommandosubstitution
3. xargs kann einem Kommando beliebig viele Parameter übergeben
4. xargs kann Parameter automatisch filtern, falls diese nicht sinnvoll übergeben werden können

**OBJECTIVE:** 1.103.4 **TYPE:** mc

Was bewirkt das folgende Kommando?

`pnpdump | tee pnp | isapnp`

1. Das Programm tee entnimmt seinen Input aus der Datei pnpdump und übergibt diesen an das Programm pnp. Das Programm pnp verarbeitet die Daten und gibt das Ergebnis an das Programm isapnp aus.
2. Das Programm isapnp speichert seinen Output in die Datei pnp, aus der die Daten wiederum mit dem Programm tee ausgelesen werden und an pnpdump übergeben werden.
3. Das Programm pnpdump leitet seinen Output an das Programm tee weiter, welches diesen in die Datei pnp speichert und gleichzeitig an das Programm isapnp weiterleitet.
4. Das Programm pnpdump leitet seinen Output an das Programm tee weiter, welches mithilfe des Skriptes

- 
- pnp Veränderungen an diesem vornimmt und das Ergebnis an isapnp übergibt
5. Die Kommandozeile ist ungültig!

**OBJECTIVE:** 1.103.5 **TYPE:** mc

Wie lautet der Befehl zum Starten eines Prozesses im Hintergrund?

1. bg | prozessaufruf
2. bg prozessaufruf
3. prozessaufruf &
4. prozessaufruf (danach Strg-Z)

**OBJECTIVE:** 1.103.5 **TYPE:** mc

Sie möchten ein im Hintergrund laufendes Programm beenden, das sich anscheinend aufgehängt hat. Wie gehen Sie vor?

1. Ermittlung der JobID mittels jobs, kill -SIGKILL PID *JobID*
2. Ermittlung der ProzessID mittels ps, kill -SIGTERM PID *ProzessID*
3. Ermittlung der ProzessID mittels ps, kill -SIGKILL PID *ProzessID*
4. Beenden des Programms mittels Strg-C
5. Ermittlung der JobID mittels jobs, fg *JobID*, Strg-Z

**OBJECTIVE:** 1.103.5 **TYPE:** mc

Sie möchten überprüfen, wie sich ein Programm im laufenden Betrieb verhält und wie viel CPU-Last es laufend erzeugt. Welches Kommando verwenden Sie?

1. jobs
2. ps fa
3. ps ax
4. top
5. taskmgr



**OBJECTIVE: 1.103.5 TYPE: mc**

Ein Prozess soll weiterlaufen, obwohl Sie vorhaben, sich von der Station abzumelden. Wie lautet der Befehl?

1. `exec Prozessaufruf`
2. `bg Prozessaufruf`
3. `nohup Prozessaufruf`
4. `top Prozessaufruf`

**OBJECTIVE: 1.103.5 TYPE: mc**

Sie haben einen im Vordergrund laufenden Prozess mit der JobID 5 mit Strg-Z angehalten, um Zugriff auf die Shell zu erhalten. Nun soll dem Prozess wieder Rechenzeit gegeben werden, ohne dass die Shell erneut dadurch besetzt wird. Wie lautet der Befehl?

1. `nohup 5`
2. `Strg-D`
3. `bg 5`
4. `kill -SIGRESTART PID 5`
5. `fg 5`

**OBJECTIVE: 1.103.6 TYPE: mcma**

Sie möchten die Priorität eines laufenden Prozesses verändern. Welche Befehle ermöglichen dies?

1. `ps`
2. `top`
3. `renice`
4. `nice`
5. `prior`

**OBJECTIVE: 1.103.6 TYPE: mc**

Sie möchten ein Programm mit einem Nice-Wert von -12 starten. Wie lautet der Befehl ?

- 
1. `top -n-12 Programmname`
  2. `nice -12 Programmname`
  3. `nice -n-12 Programmname`
  4. `renice --start -n-12 Programmname`

**OBJECTIVE:** 1.103.6 **TYPE:** mc

Sie möchten einen Dämon, der auf ihrem System läuft, die höchste Priorität zuweisen. Wie müssen Sie den Nice-Wert ändern?

1. Setzen des Nice-Wertes auf -19
2. Setzen des Nice-Wertes auf 20
3. Setzen des Nice-Wertes auf -20
4. Setzen des Nice-Wertes auf 19
5. Erhöhen des Nice-Wertes um 5

**OBJECTIVE:** 1.103.7 **TYPE:** fitb

Wie lautet der Befehl, um die Datei Userliste.txt nach den Namen "mayer", "meyer", "mayr", "maier", "meier", "meir" (M am Anfang kann groß oder klein geschrieben sein) zu suchen?

**OBJECTIVE:** 1.103.7 **TYPE:** mc

Sie möchten alle Tags in der Datei Dokument.html entfernen und als Dokument.txt abspeichern. Wie lautet der Befehl?

1. `sed -f "s/<.*>//g" Dokument.html > Dokument.txt`
2. `sed -f "s/<.*>././g" Dokument.html > Dokument.txt`
3. `sed -e "s/<.*>//g" Dokument.html > Dokument.txt`
4. `sed -e "s/<.*>//p" Dokument.html > Dokument.txt`
5. `sed -e "s/\\<.*\\>//g" Dokument.html > Dokument.txt`

**OBJECTIVE:** 1.103.7 **TYPE:** mc

Sie möchten in der Datei Dokument.html nach allen HTML-Starttags noch ein Leerzeichen (Unterstrich "\_")einfügen (Beispiel: <IrgendeinTag>Irgendwas → <IrgendeinTag>\_Irgendwas).

Wie lautet die Regular Expression?

1. `g/\(<.*>\)/\1_/s`
2. `s/<.*>/$_/g`
3. `s/\(<.*>\)/\1_/g`
4. `s/<.*>/\1_/g`

**OBJECTIVE:** 1.103.8 **TYPE:** mc

Was bewirkt der vi Befehl 9w?

1. Bewegt den Cursor zum Anfang des neunten Wortes
2. Bewegt den Cursor um neun Zeichen nach oben
3. Bewegt den Cursor zum Anfang des neunten Wortes von der gegenwärtigen Cursorposition aus
4. Bewegt den Cursor um neun Zeichen nach links
5. Fügt neun mal den Buchstaben "w" an der gegenwärtigen Cursorposition ein

**OBJECTIVE:** 1.104.1 **TYPE:** mcma

Mit welchem Befehl können Sie ein Reiser-Filesystem auf /dev/sdc2 erstellen?

1. `fdisk -t reiserfs /dev/sdc2`
2. `format --fs reiserfs /dev/sdc2`
3. `mkfs -t reiserfs /dev/sdc2`
4. `mkreiserfs /dev/sdc2`
5. `autofs --reiserfs /dev/sdc2`

**OBJECTIVE:** 1.104.1 **TYPE:** mcma

Mit welchem Befehl können Sie ein Ext3-Filesystem auf /dev/sdc2 erstellen?

1. `mkfs -tj ext3 /dev/sdc2`
2. `mkfs -j /dev/sdc2`
3. `mkfs -t ext3 /dev/sdc2`
4. `mkfs.ext3 /dev/sdc2`

---

**OBJECTIVE: 1.104.1 TYPE: mc**

Mit welchem fdisk Befehl kann man eine neue Partition erstellen?

1. a
2. p
3. n
4. o
5. x

**OBJECTIVE: 1.104.2 TYPE: mcma**

Welche Voraussetzungen sollten erfüllt sein, um das Wurzeldateisystem mit fsck zu überprüfen?

1. es sollte in den Single-User-Modus gewechselt werden
2. das Dateisystem sollte read-write-able gemountet sein
3. das Dateisystem sollte read-only gemountet sein
4. dem Programm fsck sollte beim Aufruf die höchste Prozesspriorität gegeben werden, um Unterbrechungen beim Reparaturvorgang zu vermeiden
5. das Wurzeldateisystem kann mit fsck überhaupt nicht überprüft werden, hierfür sind Sonderprogramme oder ein zweites System auf einer anderen Platte notwendig

**OBJECTIVE: 1.104.2 TYPE: mcma**

Sie möchten überprüfen, wie sehr die Partition für die Home-Verzeichnisse mit der Geräte-Datei /dev/hdc2 ausgelastet ist. Wie lautet der Befehl?

1. df -i /dev/hdc2
2. ls -l /home
3. df /home
4. df /dev/hdc2
5. df -f /dev/hdc2

**OBJECTIVE:** 1.104.2 **TYPE:** mc

Durch den Befehl `df` kommt heraus, dass die Partition `/dev/hdc2` für die Home-Verzeichnisse schon voll ist. Sie möchten nun herausfinden, welcher User vielleicht überdurchschnittlich viel Platz benötigt. Wie lautet der Befehl?

1. `du /home`
2. `du -s /home`
3. `du -s /home/*`
4. `du /home/*`

**OBJECTIVE:** 1.104.3 **TYPE:** mc

Welche Datei stellt praktisch den "Bauplan" des Dateisystems dar?

1. `/etc/mtab`
2. `/etc/fstab`
3. `/boot/fstab`
4. `/mnt/partition.table`
5. `/etc/mount.conf`

**OBJECTIVE:** 1.104.3 **TYPE:** mc

Sie haben eine CD in das CD-ROM `/dev/hdb` eingelegt und möchten diese in `/media/cdrom` mounten. Wie lauten gültige Befehle?

1. `mount /dev/cdrom /media/cdrom`
2. `mount -o ro /dev/cdrom /media/cdrom`
3. `mount -t cdfs /dev/cdrom /media/cdrom`
4. `mount /dev/cdrom /media/cdrom ro,noauto,user 0 0`
5. `mount -o iso9660,ro /dev/cdrom /media/cdrom`

**OBJECTIVE:** 1.104.3 **TYPE:** mc

Was bedeutet die Mount-Option "defaults" in `/etc/fstab`?

1. Entspricht den Einstellungen `rw`, `suid`, `dev`, `exec`, `auto`, `nouser`, `async`

- 
2. Entspricht den Einstellungen ro, suid, nodev, exec, auto, nouser, async
  3. Entspricht den Einstellungen rw, suid, dev, exec, no-auto, user, async
  4. Entspricht den Einstellungen remount, suid, dev, noexec, auto, nouser, async

**OBJECTIVE: 1.104.4 TYPE: mcma**

Was sind die Voraussetzungen, um Diskquotas auf einem Dateisystem überhaupt erst zu erstellen?

1. Der Befehl "quotacheck" muss bei jedem Systemstart abgearbeitet werden
2. Der Kernel muss Quotas unterstützen
3. Das Dateisystem muss, je nachdem wie die Mountoptionen gesetzt wurden, die Dateien quota.user und/oder quota.group enthalten
4. Das verwendete Dateisystem muss Quotas unterstützen
5. in /etc/fstab ist die Mountoption "usrquota" und/oder "grpquota" für das jeweilige Dateisystem erforderlich

**OBJECTIVE: 1.104.4 TYPE: mc**

Sie haben Quota-Einstellungen für den User "adam" festgelegt und möchten diese nun auch für "berta", "carlo" und "doris" übernehmen. Wie lautet der Befehl?

1. edquota -p adam -u berta carlo doris
2. edquota adam | edquota berta | edquota carlo | edquota doris
3. cp /home/adam/quota.user /home/\*/quota.user
4. quotaon -u adam > berta,carlo,doris

**OBJECTIVE: 1.104.4 TYPE: mc**

Mit welchem Befehl werden Diskquotas aktiviert?

1. `quotaon Dateisystem`
2. `quotaon /quota.user`
3. `quotaon -a /quota.user`
4. `quotaon -ug /etc/fstab`

**OBJECTIVE: 1.104.5 TYPE: mc**

Wie sehen die Rechte des Verzeichnisses `/tmp` nach folgendem Kommando aus?

`chmod -R 3664 /tmp`

1. `srw-rw-r--`
2. `-rw-rwsr-t`
3. `-rwsrwsrwt` für `/tmp` und alle Unterverzeichnisse
4. `drw-rwsr-t` für `/tmp` und alle Unterverzeichnisse
5. Das Kommando ist ungültig, es werden keinerlei Änderungen an den Rechten vorgenommen

**OBJECTIVE: 1.104.5 TYPE: fitb**

Sie möchten der Datei `/bin/admtool.sh` (Besitzer ist `root`) folgende Rechte zuweisen: Lese-, Schreib- und Ausführungsrecht für `root`, Lese-, Schreib und Ausführungsrecht für die Gruppe sowie Ausführungsrecht für alle anderen. Außerdem soll SUID und SGID gesetzt sein. Wie lautet der Befehl, um diese Änderungen vorzunehmen?

**OBJECTIVE: 1.104.5 TYPE: mc**

Wie sehen die Rechte der Datei `/usr/share/template` nach folgendem Kommando aus?

`chmod a-wx /usr/share/template.conf`

- 
1. Alle haben Leserecht auf die Datei  
/usr/share/template
  2. Alle haben Schreib- und Ausführungsrechte auf die  
Datei /usr/share/template
  3. Alle, außer Besitzer und dessen Primärgruppe, ver-  
lieren Schreib- und Ausführungsrecht auf die Datei  
/usr/share/template
  4. Alle verlieren Schreib- und Ausführungsrecht auf die  
Datei /usr/share/template

**OBJECTIVE:** 1.104.5 **TYPE:** mc

Sie möchten für alle neu erstellten Dateien und Verzeichnisse  
folgende Zugriffsrechte bestimmen:

`rwxr-xr—`

Wie lautet der Befehl?

1. `chmod -R 754 /`
2. `umask 023`
3. `chmod -R 023 /`
4. `chattr 754`
5. `umask -R 754`

**OBJECTIVE:** 1.104.5 **TYPE:** mcma

Sie möchten sensible Daten unwiderruflich vom System löschen.  
Wie gehen Sie vor?

1. `chattr +s Datei; rm Datei`
2. `chattr =i Datei; rm Datei`
3. `rm -r Datei`
4. ein derartiges Vorgehen ist nur auf XFS-Dateisyste-  
men möglich
5. ein derartiges Vorgehen ist nur auf EXT2-Dateisyste-  
men möglich



**OBJECTIVE: 1.104.6 TYPE: fitb**

Sie möchten als root das Verzeichnis /home/helga samt Unterverzeichnissen in Besitz nehmen, wollen die Gruppenzugehörigkeit jedoch nicht verändern. Wie lautet der Befehl?

**OBJECTIVE: 1.104.7 TYPE: mc**

Sie befinden sich im /root Verzeichnis auf der Wurzelpartition, wo Sie einen Hardlink auf die Kernel-Datei vmlinuz in der /boot Partition erstellen möchten. Wie lautet der Befehl?

1. `ln /boot/vmlinuz`
2. `ln -s /boot/vmlinuz`
3. `ln -h /boot/vmlinuz`
4. `ln -p /boot/vmlinuz`
5. Das ist nicht möglich!

**OBJECTIVE: 1.104.8 TYPE: mc**

Sie möchten alle Dateien mit der Endung .tmp von Ihrem System möglichst unaufwendig löschen. Welches Kommando würde dies ermöglichen?

1. `find / -name "*.tmp" -exec rm &_;`
2. `rm -R /*/*.tmp`
3. `find / -name "*.tmp" -exec rm {} \;`
4. `rm -r | find / *.tmp`
5. `find -name *.tmp -exec rm {} \;`

**OBJECTIVE: 1.104.8 TYPE: fitb**

Sie suchen eine mindestens 9kByte große Datei, deren Name mit "smb" anfängt und mit ".conf" aufhört. Wie lautet der Befehl?

**OBJECTIVE: 1.104.8 TYPE: mcma**

Welches Suchprogramm ist grundsätzlich vor find zu verwenden, da es nicht direkt auf der Festplatte sucht und somit Zeit spart?

1. `updatedb`

- 
2. locate
  3. grep
  4. search
  5. slocate

**OBJECTIVE:** 1.104.8 **TYPE:** mcma

Wie können ausgenommene Pfade/Dateisysteme für die Erstellung der Dateidatenbank festgelegt werden (alte sowie neue Versionen)?

1. durch spezielle Eintragungen in den Umgebungsvariablen
2. durch Mountoptionen in /etc/fstab
3. durch Kommandozeilenoptionen, die dem Programm updatedb übergeben werden
4. durch Userrechte und spezielle Dateiattribute
5. in der Datei /etc/updatedb.conf

**OBJECTIVE:** 1.104.8 **TYPE:** mc

Sie möchten das Tool "webmin" installieren, sind sich jedoch nicht sicher, ob sich dieses bereits am System befindet. Sollte es existieren, so möchten Sie auch gleich alle Pfade zu den Handbuchseiten und dgl. aufgelistet bekommen. Wie lautet der Befehl?

1. type webmin
2. which webmin
3. whatis webmin
4. whereis webmin

**OBJECTIVE:** 1.110.1 **TYPE:** mc

Beim Starten des X-Servers wird plötzlich der Bildschirm schwarz. Ihr Monitor gibt Ihnen sinngemäß folgende Fehlermeldung aus: "Out of Range". Was ist passiert?

1. Die Farbtiefe wurde für den Monitor zu hoch gewählt

2. Die Auflösung wurde für den Monitor zu hoch gewählt
3. Es wurde der falsche Chipsatz für die Grafikkarte angegeben
4. Die Grafikkarte steuert den Monitor mit einer zu hohen Synchronisationsrate an

**OBJECTIVE: 1.110.1 TYPE: mc**

Was bewirkt das Programm xset?

1. Mit xset kann man X-Clients mitteilen, welchem X-Server Sie Ihre Ausgaben schicken sollen
2. Mit xset können neue FontPath-Einträge in XF86Config aktiviert werden
3. Mit xset können dem X-Server neue Bildschirmauflösungen zugeteilt werden
4. Mit xset kann ein weiterer X-Server aktiviert werden
5. Mit xset kann man dem X-Server Ressourcendateien zuweisen

**OBJECTIVE: 1.110.1 TYPE: mc**

Sie haben eine neue TrueType Schriftdatei "Helvetica" in ein X11-Schriftenverzeichnis hineinkopiert und wollen diese nun dem System-Verfügbar machen. Wie lautet der Befehl?

1. xfs *Schriftenverzeichnis*/Helvetica.ttf
2. importfont Helvetica
3. mkfont *Schriftenverzeichnis*/Helvetica.ttf
4. mkfontdir *Schriftenverzeichnis*

**OBJECTIVE: 1.110.1 TYPE: mc**

Wie lautet der Befehl, um einen Font-Server mit einer FHS-tauglichen Konfigurationsdatei "/etc/X11/fs/config" zu starten?

1. fs -daemon
2. xfs -config /etc/X11/fs/config -daemon
3. fs /etc/X11/fs/config
4. xfs -daemon /etc/X11/fs/config

---

**OBJECTIVE: 1.110.1 TYPE: mc**

Wie erfolgt die Anbindung eines X-Servers an einen Font-Server ?

1. durch einen Eintrag nach dem Schema  
"Transportprot/FSname/Port" in die XF86Config-Sec-  
tion "Font-Server"
2. durch die Umgebungsvariable "FONT-SERVER PATH"
3. durch einen Eintrag nach dem Schema  
"Transportprot/FSname:Port" in die XF86Config-Sec-  
tion "Files"
4. durch die Startparameter "--fs  
Transportprot/FSname:Port" beim Aufruf des X-Ser-  
vers

**OBJECTIVE: 1.110.2 TYPE: mc**

Sie möchten, dass der Login-Vorgang via grafischem Login-  
Fenster über einen Display-Manager vollzogen wird. In welcher  
Datei können Sie dies aktivieren?

1. /etc/inittab
2. /etc/X11/xdm/config
3. /etc/X11/xdm/Xstartup
4. /etc/X11/xdm/Xaccess

**OBJECTIVE: 1.110.2 TYPE: mc**

Welcher Eintrag muss in der Datei xdm-config bzw. kdm-config  
stehen, damit der Displaymanager XDMCP-Pakete annimmt?

1. DisplayManager.requestPort: 0
2. DisplayManager.requestPort: 1
3. ! DisplayManager.requestPort: 0
4. # DisplayManager.requestPort: 0
5. DisplayManager.requestPort: true

**OBJECTIVE: 1.110.2 TYPE: mc**

Um welche XDMCP-Methode handelt es sich bei folgender Eingabe auf dem X-Terminal?

X :0.0 -query *Hostname*

1. QUERY
2. INDIRECT
3. CHOOSER
4. DIRECT
5. BROADCAST

**OBJECTIVE: 1.110.3 TYPE: fitb**

Sie möchten einen XTerm mit schwarzem Hintergrund und grüner Schrift starten. Wie lautet der Befehl?

**OBJECTIVE: 1.110.3 TYPE: mcma**

In welcher Datei kann jeder User seine Standardeinstellungen für X-Anwendungen festlegen?

1. /etc/X11/app-defaults
2. ~/.Xresources
3. /etc/X11/.Xresources
4. ~/.Xdefaults
5. /etc/X11/.Xdefaults

**OBJECTIVE: 1.110.3 TYPE: mc**

Mit welchem Befehl kann eine neue Ressourcendatei aktiviert werden?

1. xrdp -merge .XRessourcendatei
2. xdb -merge .XRessourcendatei
3. updatedb .XRessourcendatei
4. xrdp .XRessourcendatei
5. xset .XRessourcendatei

---

**OBJECTIVE:** 1.110.3 **TYPE:** mc

Durch welche beiden Dateien kann ein User den Startvorgang eines X-Servers beeinflussen?

1. /etc/x11/startup
2. ~/.Xsession
3. /etc/X11/xinit/xinitrc
4. ~/.xinitrc
5. /etc/X11/Xsession

**OBJECTIVE:** 1.110.3 **TYPE:** fitb

Sie möchten die Ausgaben eines XTerms auf Ihren Laptop, der sich mit der IP-Adr. 10.30.67.30 auch im Netzwerk befindet, umleiten. Wie lautet der Befehl?

---

# B

## Testfragen zur Prüfung 102

---

**OBJECTIVE:** 1.105.1 **TYPE:** mc

Mit welchem Befehl können Sie herausfinden, welche Parameter ein zu ladendes Kernel-Modul versteht?

1. modprobe -p Modulname
2. modinfo -p Modulname
3. lsmod Modulname
4. insmod -p Modulname
5. modinfo -h Modulname

**OBJECTIVE:** 1.105.1 **TYPE:** mcma

Mit welchem Befehl gelangen Sie automatisch in das Verzeichnis, mit den aktuellen Kernel-Modulen?

1. cd /etc/modules/`uname -r`
2. cd /lib/modules/`uname -r`
3. cd /usr/modules/\$uname -r
4. cd /lib/modules/\${uname -r}
5. cd /etc/modules/%uname -r%

**OBJECTIVE:** 1.105.1 **TYPE:** mc

Was kann in der Konfigurationsdatei /etc/modules.conf NICHT festgelegt werden?

1. die Kernel-Module, die beim Booten in den Kernel geladen werden sollen
2. Parameter für die zu ladenden Kernel-Module
3. Aliasnamen für die zu ladenden Kernel-Module
4. andere Kernel-Module, die vor oder nach dem Laden des gegenwärtigen Kernel-Moduls geladen werden sollen

**OBJECTIVE:** 1.105.1 **TYPE:** mcma

Mit welchen Befehlen können Kernel-Module aus dem Kernel entladen werden?

1. Während der Kernellaufzeit können keine Module entladen werden
2. umount -r
3. modprobe -r
4. uname -r
5. rmmod -r

**OBJECTIVE:** 1.105.2 **TYPE:** mc

Sie wollen ein Kernelupdate auf einen neueren Kernel durchführen. Welcher Befehl erscheint Ihnen hierfür am sinnvollsten?

1. make newconfig
2. make config
3. make xconfig
4. make oldconfig
5. make dep

**OBJECTIVE:** 1.105.2 **TYPE:** fitb

Bringen Sie den folgenden Ablauf durch Auflistung der Buchstaben in die richtige Reihenfolge.

- a) cp /usr/src/linux/System.map /boot/System.map.Kernel-Version
- b) make dep
- c) cp /usr/src/linux/arch/i386/boot/zImage /boot
- d) make modules\_install
- e) make modules
- f) make bzImage
- g) make config



---

**OBJECTIVE:** 1.105.2 **TYPE:** mcma

Wie wird ein neu erstelltes Kernel-Image aktiviert?

1. die Position des Kernel-Images muss dem Bootmanager mitgeteilt werden
2. das Kernel-Image muss auf eine Partition kopiert werden, die unter der 1024-Zylindergrenze liegt
3. in der Datei .config muss ein "active"-Flag gesetzt werden
4. das alte Kernel-Image muss durch das neue ersetzt werden
5. das Kernel-Image muss auf eine Partition kopiert werden, die beim Systemstart in den Dateisystembaum gemountet ist

**OBJECTIVE:** 1.106.1 **TYPE:** mc

Wo sollten Sie am besten Kernel-Parameter, die sich auf die zu ladenden Module beziehen, angeben?

1. in der Datei /boot/.config
2. auf dem Bootprompt des Bootmanagers
3. in der Datei /etc/lilo.conf
4. in der Datei /etc/modules.conf
5. in der Datei /boot/parameters

**OBJECTIVE:** 1.106.2 **TYPE:** mc

Ein Kernel läuft nicht ordnungsgemäß und Sie möchten überprüfen, ob vielleicht beim System-Start schon Fehler auftraten. Wie können Sie dies überprüfen?

1. durch den Befehl klogd
2. durch den Befehl dmesg
3. durch den Befehl syslogd
4. durch die Datei /var/log/messages

**OBJECTIVE:** 1.106.2 **TYPE:** mc

In welcher Datei finden Sie Informationen über die verfügbaren Kernel-Parameter?

1. /usr/local/share/linux/Documentation/kernel-Parameters.txt
2. /etc/linux/Documentation/kernel-Parameterses.txt
3. man kernelparameters
4. /boot/Documentation/kernel-Parameterses.txt
5. /usr/src/linux/Documentation/kernel-Parameterses.txt

**OBJECTIVE:** 1.106.2 **TYPE:** mc

Mit welchem Befehl fahren Sie einen Linux-Server, auf dem mehrere Benutzer eingeloggt sind, sauber herunter?

1. shutdown -h +10m
2. init 0
3. shutdown -h now
4. halt
5. suspend

**OBJECTIVE:** 1.106.2 **TYPE:** mcma

Mit welchem Befehl gelangen Sie in den Einzelbenutzermodus?

1. runlevel S
2. init 3
3. goto 3
4. init S
5. telinit s

**OBJECTIVE:** 1.106.2 **TYPE:** fitb

Welche Runlevelnummern bzw. -bezeichnungen haben eine festgelegte Bedeutung?

---

**OBJECTIVE:** 1.107.2 **TYPE:** mc

Sie möchten den Druckauftrag mit der Nummer 358 auf dem Standarddrucker aus der Druckerwarteschlange löschen. Wie lautet der Befehl?

1. `lpc clean 358`
2. `lprm 358`
3. `lpq -r 358`
4. `lpr -r 358`

**OBJECTIVE:** 1.107.3 **TYPE:** mcma

Mit welchem Befehl können Sie die Datei `/etc/passwd` in die PostScriptdatei `/root/passwd.ps` umwandeln?

1. `a2ps /etc/passwd`
2. `mpage -o /root/passwd`
3. `enscript -o /root/passwd.ps /etc/passwd`
4. `ps -o /etc/passwd /root/passwd.ps`
5. mit einem Texteditor (`vi`, `emacs`, ...)

**OBJECTIVE:** 1.107.4 **TYPE:** mc

Was bedeutet dieser `printcap`-Eintrag?

`lf=/var/spool/canon`

1. Name des Verzeichnisses, in dem die Aufträge abgespeichert werden
2. Name der Logdatei für Fehlermeldungen
3. Name der Geräte-Datei des Gerätes, an dem der Drucker angeschlossen ist
4. Name des Input-Filters, den der Drucker benutzen soll

**OBJECTIVE:** 1.108.1 **TYPE:** mc

Sie möchten die Manpage der Datei `/etc/passwd` aufrufen. Wie lautet der Befehl?

1. `man passwd`
2. `man /etc/passwd`

3. `man f passwd`
4. `man 5 passwd`
5. `apropos passwd`

**OBJECTIVE:** 1.108.1 **TYPE:** mc

Sie möchten auflisten, welche Manpages zum Thema "*browser*" vorhanden sind. Wie lautet der Befehl?

1. `man -a | find -name browser`
2. `apropos browser`
3. `man -a | grep -i 'browser'`
4. `whatis browser`
5. `whereis man -c 'browser'`

**OBJECTIVE:** 1.108.1 **TYPE:** mc

Sie haben neue Programme installiert, die /man Verzeichnisse und somit Manpages enthalten. Mit welchem Befehl aktualisieren Sie nun den Manpage-Index?

1. `man --index`
2. `manpath > $MANPATH`
3. `manpath`
4. `echo $MAMNPATH`
5. `mandb`

**OBJECTIVE:** 1.108.1 **TYPE:** mcma

Sie müssen auf die info-Seiten zurückgreifen, benötigen dazu aber erst Informationen über info selbst. Wie lauten mögliche Befehle?

1. `info --help`
2. `man info`
3. `info info`
4. `help -p info`
5. `info -?`

---

**OBJECTIVE:** 1.108.2 **TYPE:** mc

Wie lautet die Internetadresse des Linux Documentation Projects?

1. [www.linuxdoc.com](http://www.linuxdoc.com)
2. [www.linux.doc](http://www.linux.doc)
3. [www.linuxdoc.org](http://www.linuxdoc.org)
4. [www.unix.org](http://www.unix.org)
5. [www.linux.com/docs](http://www.linux.com/docs)

**OBJECTIVE:** 1.108.2 **TYPE:** mcma

Welcher der folgenden Quellen sind NICHT als Linux-Dokumentationen und/oder Hilfen geeignet?

1. [rute.sourceforge.net](http://rute.sourceforge.net)
2. [majordomo@vger.kernel.org](mailto:majordomo@vger.kernel.org)
3. [www.msn.com](http://www.msn.com)
4. [comp.os.linux.answers](http://comp.os.linux.answers)
5. [support.microsoft.com](http://support.microsoft.com)

**OBJECTIVE:** 1.108.2 **TYPE:** mcma

Welche Hilfequellen können Sie bei Linux immer kostenlos beziehen?

1. Newsgroups
2. Linux User Groups
3. HowTo's
4. Mailinglists
5. Telefon-Support

**OBJECTIVE:** 1.108.5 **TYPE:** mc

Den Inhalt welcher Datei können Sie jedem User, der sich einloggt, als Meldung Tagesmeldung übermitteln?

1. `/etc/issue`
2. `/etc/issue.net`
3. `/etc/motd`
4. `/etc/userinfo`

**OBJECTIVE:** 1.109.1 **TYPE:** mcma

Worauf ist zu achten, wenn man mit einem Script Shell-Variablen definieren will?

1. Dem Scriptaufruf muss ein Punkt und ein Leerzeichen vorangestellt werden, um die Shell zu zwingen, das Script selbst auszuführen.
2. Dem Scriptaufruf muss ein Punkt vorangestellt werden, um die Shell zu zwingen, das Script selbst auszuführen.
3. Im Script darf kein „exit“ verwendet werden
4. Im Script müssen Variablen exportiert werden, um sie an die aufrufende Shell zurückgeben zu können
5. Alle Shell-Variablen müssen vor dem Aufruf exportiert werden, um sie der Subshell und somit dem Script zugänglich zu machen

**OBJECTIVE:** 1.109.1 **TYPE:** mc

Was bewirkt das Schlüsselwort builtin in Shell-Funktionen?

1. Es gibt die Versionsdaten des Kernels zurück
2. Es gibt die Versionsdaten des Compilers für die Shell-Funktionen zurück
3. Es stellt shell-interne Befehle eventuell gleichnamigen Aliases oder Shell-Funktionen voran
4. Es exportiert Variablen in Subshells
5. Es steuert den rekursiven Aufruf von Shell-Funktionen

**OBJECTIVE:** 1.109.1 **TYPE:** mc

In welcher Datei kann die Shell-Umgebung für alle Shells des Systems konfiguriert werden?

1. ~/bashrc
2. /home/profile
3. /etc/profile
4. /root/bash\_login

---

**OBJECTIVE:** 1.109.2 **TYPE:** mcma

Welche der folgenden Scriptzeilen sind falsch?

1. `if[-r /etc/inittab]`
2. `if [-r /etc/inittab]`
3. `if [ -r /etc/inittab ]`
4. `if test -r /etc/inittab`
5. `if [ -r /etc/inittab]`

**OBJECTIVE:** 1.109.2 **TYPE:** mcma

Sie möchten alle Kommandozeilenparameter eines Shell-Skriptes mit einer Schleife abarbeiten. Welche dieser Schleifenaufrufe bewerkstelligen dies?

1. `while $@`
2. `for name in $@`
3. `while $#`
4. `while name in $@`
5. `for name`

**OBJECTIVE:** 1.109.2 **TYPE:** fitb

Was sollte in der ersten Zeile jedes Scriptes stehen, das mit der Bourne-Again-Shell ausgeführt werden soll, um dessen Ausführung unter jeder Shell zu gewährleisten?

**OBJECTIVE:** 1.111.1 **TYPE:** mc

Mit welchem Kommando wird auch ein HOME-Verzeichnis für einen anzulegenden User mitangelegt?

1. `userhome`
2. `usermod -h`
3. `useradd`
4. `useradd -m`

**OBJECTIVE: 1.111.1 TYPE: mc**

Was bewirkt das Programm chfn?

1. Damit können User ihre Gruppenzugehörigkeit ändern
2. Damit kann der Systemverwalter die Kommentarfelder der Userinformationen editieren
3. Damit können die User sowie der Systemverwalter die entsprechenden Kommentarfelder der Userinformationen ändern
4. Damit kann der Systemverwalter die Startshells der User ändern
5. Damit kann der Systemverwalter die Passwort-Gültigkeits-Informationen editieren

**OBJECTIVE: 1.111.1 TYPE: mc**

Was bewirkt das Programm pwconv?

1. Es überprüft die Konsistenz der Passwortdateien
2. Es konvertiert die alte /etc/passwd-Datei in eine modernere passwd/shadow Kombination
3. Es ändert den Verschlüsselungsalgorithmus zur Speicherung der Passwörter (z. B. MD5 zu Blowfish)
4. Es konvertiert eine modernere passwd/shadow Kombination in die alte /etc/passwd-Datei
5. Es liest das Passwort eines bestimmten Users und gibt es im Klartext aus

**OBJECTIVE: 1.111.1 TYPE: mc**

Wie kann ein User sein Passwort ändern?

1. pwconv
2. pw
3. passwd
4. setpass
5. pass



---

**OBJECTIVE:** 1.111.2 **TYPE:** mc

Was unterscheidet Login- und Nicht-Login-Shell voneinander?

1. Eine Nicht-Login-Shell arbeitet keine Startdateien ab
2. Eine Nicht-Login-Shell wird wesentlich schneller aufgerufen, da Sie alle wichtigen Umgebungsvariablen von der Login-Shell erbt
3. Eine Login-Shell wird wesentlich schneller aufgerufen, da Sie alle wichtigen Umgebungsvariablen von der Nicht-Login-Shell erbt
4. Eine Nicht-Login-Shell erbt alle Umgebungsvariablen der Login-Shell beim Aufruf
5. Eine Login-Shell erbt alle mit export deklarierten Umgebungsvariablen der Nicht-Login Shell beim Aufruf

**OBJECTIVE:** 1.111.2 **TYPE:** mc

Welche der folgenden Dateien konfiguriert (abgesehen vom Variablen-Export) eine Nicht-Login-Shell beim Start?

1. /etc/profile
2. ~/bash\_login
3. ~/.profile
4. ~/bashrc
5. ~/bash\_nologin

**OBJECTIVE:** 1.111.2 **TYPE:** mc

Was beinhaltet das Verzeichnis „/etc/skel“?

1. Das HOME-Verzeichnis von "root"
2. Das HOME-Verzeichnis des "system"-users
3. Das HOME-Verzeichnis eines Muster-Users
4. Einige grundlegende Dateien, die in das HOME-Verzeichnis jedes Users "verlinkt" werden

**OBJECTIVE: 1.111.3 TYPE: mc**

Welche Beschreibung trifft auf die Syslog-Priorität „emerg“ zu?

1. Nachricht, die sofortiges Eingreifen erforderlich macht
2. Nachricht über kritische Situation, die aber vom System bewältigt werden konnte
3. Fehlermeldung jeglicher Art aus dem laufenden Betrieb
4. Die letzte Meldung vor einem Absturz

**OBJECTIVE: 1.111.3 TYPE: mcma**

Welche Möglichkeiten bietet Linux, um Logdateien in Echtzeit auszugeben?

1. tail -f
2. logrotate
3. syslog -f
4. less -F
5. debug

**OBJECTIVE: 1.111.3 TYPE: mcma**

Wie wird das Programm logrotate sinnvoller Weise verwendet?

1. mittels einer Konfigurationsdatei, mit den Rotationskriterien
2. mittels Kommandozeilenparametern, die die Rotationskriterien festlegen
3. als Dämon, in Echtzeit
4. als cron-Job
5. logrotate wird automatisch vom syslogd-Dämon aufgerufen, wenn dieser feststellt, dass die Rotationskriterien erfüllt sind

---

**OBJECTIVE: 1.111.4 TYPE: mc**

Was passiert, wenn sowohl die /etc/cron.allow als auch die /etc/cron.deny Datei nicht existiert?

1. Alle Benutzer dürfen eigene Crontabs definieren
2. Alle Benutzer erhalten Zugriff auf die Datei /etc/crontab
3. Kein Benutzer darf Crontabs definieren
4. Kein Benutzer außer "root" darf Crontabs definieren
5. Es verändert sich nichts

**OBJECTIVE: 1.111.4 TYPE: mc**

Was bewirkt dieser crontab-Eintrag?

`* * 20 * 1 /usr/bin/backup`

1. Das Kommando /usr/bin/backup wird an jedem 20. im Monat und jedem Montag um 0:00 Uhr ausgeführt
2. Das Kommando /usr/bin/backup wird an jedem Montag, den 20., um 0:00 Uhr ausgeführt
3. Das Kommando /usr/bin/backup wird an jedem Montag, den 20., um 24:00 Uhr ausgeführt
4. Das Kommando /usr/bin/backup wird an jedem 20. Januar im Jahr um 24:00 Uhr ausgeführt

**OBJECTIVE: 1.111.4 TYPE: mc**

Was unterscheidet „cron“ und „anacron“ voneinander?

1. cron kann auf bestimmte Ereignisse reagieren und dementsprechend Jobs starten
2. cron ist ein Dämon-Prozess, anacron ein einfaches Programm
3. cron führt die definierten Jobs nur in ganz bestimmten Intervallen durch, anacron nach bestimmten Zeitspannen
4. anacron führt auch dann Jobs aus, wenn der Computer nicht läuft
5. anacron ist für Usercrontabs zuständig

**OBJECTIVE: 1.111.4 TYPE: mc**

Wo werden at-Jobs definiert?

1. in der Datei /etc/at.allow
2. in der Datei /etc/at.deny
3. in der Datei /etc/atab
4. durch das Programm "at" in einem Spoolsystem

**OBJECTIVE: 1.111.5 TYPE: mc**

Sie möchten ein inkrementelles Backup basierend auf einem monatlichen Vollbackup erstellen. Wie lautet der Befehl?

1. `dump -0u -f /dev/tape`
2. `dd -1 -f /dev/tape`
3. `dump -1u -f /dev/tape`
4. `bu -2u -f /dev/tape`

**OBJECTIVE: 1.111.5 TYPE: mc**

Sie möchten sämtliche Dateien mit der Endung .conf auf Ihrem Rechner in ein Archiv sichern. Welcher Befehl ermöglicht Ihnen dies?

1. `find / -name *.conf | tar -f backup`
2. `dump -0ua -f backup *.conf`
3. `find / -name *.conf | cpio -o > backup`
4. `rescue / -b -t "text/conf"`

**OBJECTIVE: 1.111.5 TYPE: mc**

Was bewirkt dieser Befehl?

`dd if=/dev/hda1 of=data.img bs=512 count=1`

1. Es schreibt die ersten 512 Blöcke der ersten Plattenpartition von /dev/hda in die Datei data.img
2. Es sichert die ersten 512 Bytes der Datei data.img auf Band

- 
3. Es sichert alle Daten der ersten Plattenpartition von /dev/hda in die Datei data.img
  4. Es schreibt den "Master Boot Record" der ersten Plattenpartition von /dev/hda in die Datei data.img

**OBJECTIVE: 1.111.6 TYPE: mc**

Was bewirkt dieses Kommando?

`date "%m%d%H%M"`

1. Es gibt die Systemzeit in Monat, Tag, Stunden, Minuten aus
2. Es gibt die Hardwarezeit in Monat, Tag, Stunden, Minuten aus
3. Die Systemzeit übernimmt Monat, Tag, Stunden, Minuten der Hardwareuhr
4. Die Hardwareuhr übernimmt Monat, Tag, Stunden, Minuten der Systemzeit
5. Fehlermeldung: ungültiges Datum ' %m%d%H%M'

**OBJECTIVE: 1.111.6 TYPE: mc**

Was unterscheidet das Kommando „date“ vom Kommando „hwclock“?

1. date gibt das Datum aus, während hwclock die Zeit ausgibt
2. date gibt Datum und Uhrzeit aus, während hwclock zum Einstellen der Zeit dient
3. date ist für die Systemzeit zuständig, während hwclock für die BIOS-Uhr zuständig ist
4. date ist für die CMOS-Uhr zuständig, während hwclock für die Systemzeit zuständig ist

**OBJECTIVE: 1.111.6 TYPE: mcma**

Wann muss das Programm ntpdate dem Dämon ntpd vorgezogen werden?

1. wenn die lokale Zeit wahrscheinlich um mehr als 1000 Sekunden verstellt ist
2. wenn die lokale Zeit wahrscheinlich um mehr als 100 Sekunden verstellt ist
3. wenn der Rechner keine Netzwerkanbindung besitzt
4. wenn es sich um einen reinen Client Rechner handelt, der keine weiteren Clients bedienen soll
5. der Dämon ist um einiges vielseitiger und deshalb immer dem Programm ntpdate vorzuziehen

**OBJECTIVE: 1.111.6 TYPE: mc**

Was enthält die Datei /etc/ntp.drift?

1. die Zeitserver, die vom NTP-Dämon abgefragt werden sollen
2. die komplette Konfiguration für den NTP-Dämon
3. die Abweichung der lokalen Zeit in Millisekunden
4. die Abweichung der lokalen Zeit in PPM (Parts per Million)
5. die Abgleichungsperiode mit den Zeitservern

**OBJECTIVE: 1.112.1 TYPE: mcma**

Welche dieser Adressbereiche werden im Internet NICHT geroutet?

1. 10.30.0.0 bis 10.30.255.0
2. 127.16.0.0 bis 127.31.0.0
3. 172.16.0.0 bis 172.31.0.0
4. 192.168.0.0 bis 192.168.255.0
5. 196.162.0.0 bis 196.162.255.0

---

**OBJECTIVE:** 1.112.1 **TYPE:** mc

Vervollständigen Sie diese Aussage: TCP arbeitet ..., UDP arbeitet ...

.

1. verbindungslos, verbindungsorientiert
2. verbindungsorientiert, verbindungslos
3. auf der Transportschicht, auf der Vermittlungsschicht
4. auf der Transportschicht, auf der Anwendungsschicht
5. zusammen mit IP, als selbstständiges Protokoll

**OBJECTIVE:** 1.112.1 **TYPE:** mc

Welcher Dienst benutzt den Port 22?

1. telnet
2. ftp
3. ssh
4. xftp
5. smtp

**OBJECTIVE:** 1.112.1 **TYPE:** mc

Wozu dient das Programm "dig"?

1. für DNS-Serverabfragen
2. zur Terminalemulation
3. zur Ermittlung, ob ein Host erreichbar ist
4. um UDP-Verbindungen aufzubauen

**OBJECTIVE:** 1.112.3 **TYPE:** fitb

Wie lautet der Befehl, um das Interface "eth0" einzuschalten?

**OBJECTIVE:** 1.112.3 **TYPE:** mc

Sie möchten eine Routing Table zum Gateway 192.168.150.40 erstellen. Wie lautet der Befehl?

1. `addroute -gw 192.168.150.40`
2. `route add default gw 192.168.150.40`
3. `route add gw 192.168.150.40`
4. `route --gateway 192.168.150.40`

**OBJECTIVE:** 1.112.3 **TYPE:** mc

Was bewirkt dieses Kommando?

`netstat -a`

1. Es zeigt die Netzwerkstatistik für Ethernet-Interfaces an
2. Es zeigt die lokalen Routing-Tables an
3. Es zeigt die Netzwerkstatistik für alle Protokolle an
4. Es zeigt die Netzwerkstatistik für alle Interfaces an
5. Es zeigt die Netzwerkstatistik für das TCP-Protokoll an

**OBJECTIVE:** 1.112.3 **TYPE:** mcma

Welche Vorteile bringt die Anbindung an einen DHCP-Server?

1. einen größeren Adressbereich, weshalb mehr Clients eine Anbindung erhalten können als durch statische IP-Adressen
2. die Netzwerk-Interfaces des Clients müssen nicht mehr konfiguriert werden
3. die Netzwerk-Interfaces des Clients müssen nicht aktiviert sein, sie werden erst nach Bedarf aktiviert -> Sicherheit (!)
4. nicht verwendete IP-Adressen stehen bei Bedarf anderen Clients zur Verfügung



---

**OBJECTIVE: 1.112.3 TYPE: mc**

In welcher Datei werden DNS Namen des lokalen Netzes mit ihren IP-Adressen verknüpft?

1. /etc/hosts
2. /etc/hostname
3. /etc/localhosts
4. /etc/host.conf

**OBJECTIVE: 1.112.3 TYPE: mc**

Wie lässt sich die Datei "/etc/nsswitch.conf" am besten beschreiben?

1. sie dient dazu, einem Rechner mehrere IP-Adressen zuzuweisen, zwischen denen dann "geswitcht" werden kann
2. sie dient dazu, einen Linux-Rechner als Netzwerk-Switch zu konfigurieren
3. sie legt fest, ob diverse Systemeinstellungen lokal oder aus dem Netz bezogen werden
4. in ihr werden alle verfügbaren DNS-Server festgelegt, und nach Bedarf verwendet

**OBJECTIVE: 1.112.3 TYPE: mc**

Was bewirkt dieser Befehl?

`tcpdump -i eth0`

1. Es gibt die Paketheader aller Pakete aus, die das Interface eth0 erreichen
2. Es gibt die Paketheader aller Pakete aus, die an das Interface eth0 gerichtet waren
3. Es gibt alle Pakete, die an das Interface eth0 gerichtet waren, aus
4. Es erstellt ein Systembackup und schreibt dieses ins Netzwerk

**OBJECTIVE: 1.112.4 TYPE: mc**

Wozu dient das PPP Protokoll?

1. Zur Realisierung eines Netzwerks über ein Modem
2. Zur Realisierung von File-Sharing Diensten
3. Zum Holen von Mails von einem Mail-Server
4. Zur Realisierung einer IP-Verbindung zwischen maximal zwei Kommunikationspartnern

**OBJECTIVE: 1.112.4 TYPE: mc**

Was unterscheidet "wvdial" und "pppd" voneinander?

1. "wvdial" ist der PPP-Client, "pppd" der PPP-Server
2. "wvdial" dient zum Aufbau einer Modemverbindung, durch die dann "pppd"-Dämonen kommunizieren können
3. "pppd" dient zum Aufbau einer Modemverbindung, durch die dann "wvdial"-Dämonen kommunizieren können
4. "wvdial" ist ein Frontend für den "pppd"-Dämon

**OBJECTIVE: 1.112.4 TYPE: mc**

Wie heißt die Option, die den "pppd"-Dämon dazu veranlasst, nach einem Verbindungsabbruch die Verbindung neu aufzubauen?

1. auto
2. reconnect
3. maxfail 0
4. idle
5. persist

---

**OBJECTIVE:** 1.113.1 **TYPE:** mc

Welchen Socket-Typ benutzt das TCP-Protokoll in den meisten Fällen?

1. dgram
2. seqpacket
3. raw
4. stream
5. rdm

**OBJECTIVE:** 1.113.1 **TYPE:** mc

Wofür ist der TCP-Dämon zuständig?

1. er stellt dem System das TCP-Protokoll zur Verfügung
2. er verwaltet die TCP-Ports des Systems
3. er regelt, welcher Host welchen Dienst verwenden darf
4. er filtert TCP-Pakete nach bestimmten Kriterien

**OBJECTIVE:** 1.113.1 **TYPE:** mc

Wo werden die von inetd zu verwaltenden Dienste definiert?

1. in einer grafischen Oberfläche, die inetd über das Web anbietet
2. in der Datei /etc/inetd.conf
3. inetd enthält bereits alle möglichen Dienste und benötigt keinerlei Konfiguration
4. in der Datei /etc/services

**OBJECTIVE:** 1.113.1 **TYPE:** mc

Was bedeutet das Schlüsselwort instances in den Sektionen der xinetd Konfigurationsdatei?

1. wie viele Instanzen des Servers gleichzeitig laufen dürfen
2. wie viele Instanzen sich den Dienst aufteilen (multi-threaded Server)
3. wie oft ein Serveraufruf von xinetd fehlschlagen darf
4. wie viele Ports der Server für seine Zwecke nutzen darf

**OBJECTIVE: 1.113.2 TYPE: mc**

Wann muss der sendmail-Dämon permanent im Speicher laufen?

1. wenn auch E-Mails von sendmail versendet werden
2. wenn auch E-Mails von sendmail empfangen werden
3. wenn jederzeit ein MUA-Programm gestartet werden können soll
4. Wenn das IMAP oder das POP3-Protokoll eingesetzt werden soll
5. Wenn der Rechner einen temporären Internetzugang besitzt

**OBJECTIVE: 1.113.2 TYPE: mcma**

Mit welchen Möglichkeiten könnte der Systemverwalter seine E-Mails automatisch an seinen normalen Login-Namen schicken lassen?

1. durch einen Eintrag in `/etc/aliases: root: Username`
2. durch Angabe des Usernames in der `.forward`-Datei im HOME-Verzeichnis von root
3. durch einen Eintrag in der Datei `/etc/sendmail.cf`
4. durch Angabe der Mailqueue des Users in der `.forward`-Datei im HOME-Verzeichnis von root

**OBJECTIVE: 1.113.2 TYPE: mcma**

Was muss erfüllt werden, damit sendmail Mails ins Internet versenden kann?

1. es liegt eine korrekte DNS-Installation auf dem Rechner vor
2. der sendmail- Dämon läuft permanent im Speicher
3. es existiert ein Smarthost im Netz, an den die Mails weitergegeben werden können
4. es existiert eine `".forward"` Datei im HOME-Verzeichnis des Users, der die Mail versenden will

---

**OBJECTIVE: 1.113.2 TYPE: mc**

Was ist bei allen Dateien im Verzeichnis /etc/mail zu beachten?

1. sie dürfen nicht geändert werden
2. sie liegen in einem verschlüsselten Datenformat vor
3. sie müssen Schreib-Rechte für den User "sendmail" aufweisen
4. sie müssen Lese-Rechte für den User "sendmail" aufweisen
5. sie müssen in einem speziellen Datenbankformat vorliegen, damit sendmail sie lesen kann

**OBJECTIVE: 1.113.3 TYPE: mc**

Was muss nach jeder Änderung in der Datei httpd.conf erfolgen?

1. ein Neustart des Systems
2. ein Neustart des initd-Dämons
3. eine Konvertierung der Konfigurationsdatei in ein Datenbankformat
4. ein Neustart des httpd-Dämons

**OBJECTIVE: 1.113.3 TYPE: mc**

Wo werden Webseiten gespeichert, die Apache der Allgemeinheit anbieten soll?

1. im Verzeichnis /etc/httpd/htdocs
2. im Verzeichnis /srv/www/htdocs
3. im Verzeichnis /usr/local/httpd/htdocs
4. siehe "DocumentRoot" in der Datei httpd.conf

**OBJECTIVE: 1.113.3 TYPE: mc**

Was beschreibt die globale Einstellung "KeepAlive" des httpd-Dämons?

1. sie legt die "TTL" (Time to Live) der versendeten IP-Pakete fest

2. sie legt fest, wie lange ein Child-Prozess des Servers laufen darf, falls keine Anfragen zur Bearbeitung eintreffen
3. sie legt fest, wie viele Anfragen hintereinander über eine Verbindung bearbeitet werden dürfen, ohne dass ein erneuter TCP-Handshake erfolgen muss
4. sie legt fest, ob mehrere Anfragen hintereinander über eine Verbindung bearbeitet werden dürfen, ohne dass ein erneuter TCP-Handshake erfolgen muss
5. sie legt fest, wie lange das Betriebssystem den Web-Server laufen lässt, wenn dieser keine Lebenszeichen von sich gibt

**OBJECTIVE:** 1.113.3 **TYPE:** mc

Sie möchten den Apache-Web-Server im laufenden Betrieb neu starten, ohne dass bestehende Verbindungen gelöst werden. Wie lautet der Befehl?

1. `rcapache try-restart`
2. `rcapache graceful`
3. `apachectl ripeful`
4. `apachectl graceful`
5. `httpd --gr -f /etc/httpd/httpd.conf`

**OBJECTIVE:** 1.113.4 **TYPE:** mcma

Sie möchten die Freigabe `wks13a:/usr/public` in das Verzeichnis `/mnt/wks13a` mounten. Wie lautet der Befehl?

1. `smbmount wks13a:/usr/public /mnt/wks13a`
2. `mount -t nfs wks13a:/usr/public /mnt/wks13a`
3. `mount -t smbfs wks13a:/usr/public /mnt/wks13a`
4. `mount wks13a:/usr/public /mnt/wks13a`

---

**OBJECTIVE:** 1.113.4 **TYPE:** mcma

Welche Bedingungen müssen erfüllt sein, damit man selbst mit NFS Verzeichnisse freigeben kann?

1. der rpc.nfsd-Dämon muss laufen
2. der RPC-Portmapper muss laufen
3. der inetd-Dämon muss laufen
4. der smbd-Dämon muss laufen
5. der rpc.mountd-Dämon muss laufen

**OBJECTIVE:** 1.113.4 **TYPE:** mc

Was beschreibt das "Squashing"?

1. eine Technik, mit der Konflikte zwischen den UserID's von Rechnern, die sich mit NFS verbinden, vermieden werden
2. eine Technik, mit der Linux-Rechner durch spezielle Einträge in der Datei /etc/exports über Adresskürzel ansprechbar werden
3. eine Technik, mit der Hacker die UserID-Schwachstelle des NFS-Systems ausnutzen, um sich "root"-Rechte zu verschaffen
4. die Art und Weise, wie ein NFS-Server eine Verbindungsabfrage zurückweist (z. B. "silentmode")

**OBJECTIVE:** 1.113.4 **TYPE:** mc

In welcher Sektion der Datei /etc/smb.conf können die HOME-Verzeichnisse der User zur Verwendung in Windows-Netzen zugänglich gemacht werden?

1. in der Sektion "global"
2. in der Sektion "homes"
3. in der Sektion "shares"
4. in der Sektion "public"
5. in der Sektion "users"

**OBJECTIVE: 1.113.5 TYPE: mc**

Was ist der Unterschied zwischen einem normalen und einem "caching-only"-DNS-Server?

1. der "caching-only"-DNS-Server kann nur Adressen in Namen umwandeln ("dns-lookup")
2. der "caching-only"-DNS-Server kann nur Namen in Adressen umwandeln ("dns-reverse-lookup")
3. der "caching-only"-DNS-Server arbeitet nicht mit den "root-servern" zusammen, sondern arbeitet nur mit der Datei /etc/hosts
4. der "caching-only"-DNS-Server schickt DNS-Anfragen direkt weiter an den nächsten "echten" DNS-Server oder arbeitet mit den Informationen, die er im Cache gespeichert hat

**OBJECTIVE: 1.113.5 TYPE: mcma**

Was unterscheidet die beiden DNS-Servergenerationen BIND4 und BIND8 voneinander?

1. "BIND4" kann nur mit 4 IP-Adressbytes umgehen, während "BIND8" für "IPv6" entwickelt wurde und maximal sogar mit 8 IP-Adressbytes umgehen kann
2. die Konfigurationsdatei von "BIND4" hat ein eigenes Format, während die von "BIND8" in einem C-Code ähnlichen Format geschrieben wird
3. die Konfigurationsdatei von "BIND4" liegt in einem C-Code ähnlichen Format vor, während die von "BIND8" in "XML" geschrieben wird
4. die Dateien im Verzeichnis /var/named unterscheiden sich in den beiden Versionen wesentlich voneinander
5. "BIND4" wird mit den Programmen "named.restart" und "named.reload" gesteuert, während "BIND8" nur mit einem HUP-Signal dazu gezwungen werden kann, seine Konfigurationsdatei neu einzulesen



---

**OBJECTIVE: 1.113.5 TYPE: mc**

Was wird mit dem "BIND4"-Konfigurationsparameter "directory" definiert?

1. die Datei, die die DNS-Servereinträge enthält
2. das Verzeichnis, das weitere Konfigurationsdateien des DNS-Servers enthält
3. das Verzeichnis, auf das sich die Konfigurationsdatei im Weiteren bezieht
4. die Domain, die der DNS-Server verwaltet
5. ein Informationsverzeichnis im Internet, aus dem alle DNS-Server des Internets ihre Informationen beziehen

**OBJECTIVE: 1.113.5 TYPE: mc**

Welcher Teil dieser URL gibt die übergeordnete Domain an, in der sich der Server befindet?

`http://www.shop.bhv.co.at`

1. `http:`
2. `.at`
3. `.co`
4. `//www`
5. `.shop`

**OBJECTIVE: 1.113.7 TYPE: mc**

Was steht in der Datei `/etc/nologin`?

1. Informationen über die Clients, die sich NICHT am Server anmelden/einloggen dürfen
2. eine Meldung, die jedem User übermittelt wird, wenn dieser versucht, sich auf dem Server einzuloggen
3. eine Meldung, die einem User (außer root) übermittelt wird, wenn dieser versucht, sich auf dem Server einzuloggen
4. eine Fehlermeldung, die an einen Client übermittelt wird, falls der Login-Vorgang fehlschlägt

**OBJECTIVE: 1.113.7 TYPE: mc**

In welcher Datei wird der sshd-Dämon konfiguriert?

1. /etc/sshd.conf
2. /etc/conf/sshd
3. /etc/sshd/config
4. /etc/sshd/sshd\_config

**OBJECTIVE: 1.113.7 TYPE: mc**

Welche Schlüssel werden bei einer Clientabfrage vom Server an den Client geschickt?

1. Host-Key
2. Server-Key
3. Secret-Key
4. Public-Key
5. Private-Key

**OBJECTIVE: 1.113.7 TYPE: mc**

Wie lautet der Befehl um ein 1024 Bit RSA-Schlüsselpaar zu erzeugen?

1. keygen -t rsa
2. ssh -k -w 1024 -t rsa
3. ssh-keygen -t rsa
4. rand --key 1024 -t rsa

**OBJECTIVE: 1.114.1 TYPE: mc**

Wie können Sie systemweit Dateien mit speziellen Benutzerrechten (Bsp.: SUID und GUID gesetzt) auffinden?

1. find / -perm 6000
2. ls / | grep -perm 6\*\*\*
3. find / -perm -6000
4. ls / -perm 6\*\*\*
5. fileagent --getpermission "SUID || GUID"

---

**OBJECTIVE:** 1.114.1 **TYPE:** mc

Was bedeutet folgende ipchains-Anweisung

```
ipchains -A output -i eth0 -y -p TCP -s any/0 -d 10.209.13.4/0 -j  
DENY
```

1. alle Versuche eines jeden Rechners im Netz, eine TCP-Verbindung zu irgendeinem Port unseres Rechners aufzubauen, werden wortlos zurückgewiesen
2. alle Versuche unseres Rechners, eine TCP-Verbindung zu irgendeinem Port eines jeden Rechners im Netz aufzubauen, werden wortlos zurückgewiesen
3. alle Versuche eines jeden Rechners im Netz, eine TCP-Verbindung zu irgendeinem Port unseres Rechners aufzubauen, werden mit einer Fehlermeldung zurückgewiesen
4. alle Versuche unseres Rechners, eine TCP-Verbindung zu irgendeinem Port eines jeden Rechners im Netz aufzubauen, werden mit einer Fehlermeldung zurückgewiesen
5. alle von unserem Rechner ausgehenden Verbindungsversuche zu irgendeinem Port irgendeines Rechners im Netz werden wortlos zurückgewiesen

**OBJECTIVE:** 1.114.1 **TYPE:** mcma

Welche Grundeinstellungen sollten immer am Anfang jeder "ipchains-Definition" stehen?

1. ipchains -P *[input | output | forward]* REJECT
2. ipchains -P *[input | output | forward]* ACCEPT
3. ipchains -P *[input | output | forward]* DENY
4. ipchains -F

**OBJECTIVE: 1.114.2 TYPE: mc**

Welcher Herkunft sind Syslog-Meldungen, die die ipchains-Firewall betreffen?

1. authpriv
2. auth
3. kern
4. daemon
5. local0

**OBJECTIVE: 1.114.2 TYPE: mc**

Welche dieser Dateien sollten Sie zuerst erstellen, wenn schwerwiegendere Serverwartungsarbeiten bevorstehen?

1. /etc/fstab
2. /etc/nologin
3. /etc/motd
4. /etc/shadow

**OBJECTIVE: 1.114.2 TYPE: mc**

Wie werden Stand-Alone gestartet bzw. gestoppt?

1. durch GUI' s
2. durch "rcxxx" Scripte
3. durch init.d Scripte
4. durch die jeweiligen Dämon-Aufrufe

**OBJECTIVE: 1.114.3 TYPE: fitb**

Mit welchem shellinternen Kommando können von der Shell benötigte System-Ressourcen limitiert werden (nur Kommando)?

---

## Schlagwortverzeichnis

---

---

### &

& 100

---

### .

.bash\_login 267  
.bash\_logout 267  
.bash\_rc 27  
.in-addr-arpa 359  
.rhosts 401  
.Xdefaults 245

---

### /

/boot/grub/grub.conf 148  
/etc/aliases 470  
/etc/anacrontab 289  
/etc/apsfilter 250  
/etc/apt/sources.list 200  
/etc/at.allow 286, 462  
/etc/at.deny 286, 462  
/etc/conf.modules 180  
/etc/cron.allow 461  
/etc/cron.deny 461  
/etc/crontab 286  
/etc/dpkg/dpkg.cfg 198  
/etc/exports 377  
/etc/fstab 438  
/etc/group 69  
/etc/gshadow 70  
/etc/host.conf 323  
/etc/HOSTNAME 349  
/etc/hosts 348  
/etc/hosts.allow 338  
/etc/hosts.deny 338  
/etc/hosts.equiv 401  
/etc/hosts.lpd 249  
/etc/hotplug 222

/etc/inittab 150  
/etc/isapnp.conf 214  
/etc/issue 156, 455  
/etc/issue.net 157  
/etc/ld.co.cache 183  
/etc/ld.so.conf 183  
/etc/lilo.conf 143  
/etc/modules.conf 180  
/etc/motd 455  
/etc/named.boot 357  
/etc/named.conf 357  
/etc/networks 348  
/etc/nologin 475  
/etc/nsswitch.conf 347, 467  
/etc/ntp.conf 166  
/etc/ntp.drift 464  
/etc/pam.d/\* 397  
/etc/passwd 73  
/etc/printcap 247, 252  
/etc/resolv.conf 322  
/etc/security/limits.conf 398  
/etc/services 116, 318  
/etc/skel 459  
/etc/smb.conf 473  
/etc/ssh/ssh\_config 412  
/etc/syslog.conf 158  
/etc/usbmgr/usbmgr.conf 222  
/etc/vsftpd.conf 335  
/etc/X11 226  
/home 67  
/usr/local/httpd 372  
/usr/X11R6/lib/X11 226  
/var/lock-Verzeichnis 420  
/var/log/messages 160  
/var/spool/cron/crontabs 287

---

### 1

100BaseT 308  
100BaseT 308  
10Base5 306  
10BaseT 307

---

## **A**

absoluter Pfad 50  
Accountname 67  
alias 49  
Aliasdatenbank 369  
anacron 289, 461  
Apache 372  
apachectl 192, 375, 472  
apropos 92  
Apsfilter 250  
apt 199  
apt-cache 200  
apt-get 201, 425  
ARP 311  
ARPA-Net 303  
at 284, 462  
atrm 285  
audio 216  
ausschneiden Siehe cut  
Automatisieren 283

---

## **B**

Backup 462  
Backup-Arten 291  
Banner 157  
bash 18, 428  
batch 286  
Baudrate 210  
Benutzerrechte 77, 79  
Benutzerverwaltung 67  
bg 433  
Binärzahl 314  
BIND 355  
BIND4 475  
BIOS 12, 143, 207  
Blockgröße 30  
blockorientiert 29  
Bootmanager 143  
Bootprozess 143, 149  
Bootvorgang 209  
Broadcast 313  
bunzip2 187

bzcat 187  
bzImage 176  
bzip2 186, 421

---

## **C**

case 279  
cat 60  
cd 24  
CD-ROM 38  
chage 74  
change shell Siehe chsh  
chattr 85  
Cheapernet 307  
chgrp 80  
chmod 80, 440  
chown 80  
chroot 337  
chsh 28  
CIDR 316  
CNAME 361  
Compiler 169  
Computerarchitektur 208  
copy Siehe cp  
cp 107  
cpio 295, 298  
crond 286  
cron-daemon Siehe crond  
crontab 287, 461  
Crypt 394  
CSMA/CD 310  
cut 130, 430

---

## **D**

date 27, 163, 463  
Dateirechte 79  
Dateiumleitung 128  
Datenblöcke 111  
dd 462  
debconf 206  
Debian Paketverwaltung 197  
Defaultboot 148

---

Defaultroute 316  
depmod 180  
Desktop-Manager 15  
df 437  
DFÜ 211  
DHCP 350  
dhcpd 352  
DHCP-Server 466  
dig 346, 465  
Disasterrecovery 291  
Diskquotas 439  
DMA 208  
DMZ 405  
DNS 304  
DNS-Server 324, 345, 355, 369,  
474  
DoD 303  
dpkg 198, 424  
Drucken 247  
Drucker 247  
Druckerqueue 259  
Druckerwarteschlange 453  
du 438  
Dualbootkonfiguration 143  
Dualbootsystem 299  
dump 300

---

## **E**

echo 20  
edquota 439  
Eingabeaufforderung 19  
Eingabe-Verarbeitung-Ausgabe  
Siehe EVA  
EISA 207  
elif then-Anweisung 270  
E-Mail 470  
entschlüsseln 395  
env 19  
eth0 465  
Ethernet 306  
EVA 127  
exec 103  
exportfs 379

ext2 29, 31  
ext3 29, 34

---

## **F**

FDDI 308  
fdisk 437  
FDISK 39  
fg 103  
file 59  
Filesystem 144  
find 112, 476  
Firewall 401  
Firewall-Skript 407  
fonts.dir 241  
Fontserver 235  
for do-Anweisung 275  
FQDN 323  
Freigaben 377  
fsck 42, 437  
fstab 63  
ftp 333, 338  
FTP Server 333  
FTP-Client 341  
FTP-Server 340

---

## **G**

Gateway 466  
GDI-Drucker 262  
Geräte-Dateien 36  
ghostscript 257  
GID 77  
Glasfasertechnologien 308  
GNOME 242  
gpasswd 70  
Grafikchip 224  
Grafikkarte 227  
Grand Unified Bootloader  
Siehe GRUB  
graphische Oberfläche 99  
groupadd 69  
groupdel 75

---

groupmod 76  
grpconv 77  
GRUB 143  
grub-install 148  
Gruppendiskriptor 34  
gunzip 185

---

## **H**

Hardlink 110, 442  
Hardware 167, 211  
Hardware Uhr 166  
hda 37  
head 115  
Hilfe 89  
Hintergrund-Prozess 100  
HISTFILESIZE 28  
HISTSIZ 28  
HOME 50  
host 345  
hostname 349  
hotplug 222  
HOWTO 93, 208  
httpd.conf 372, 471  
hwclock 166

---

## **I**

I/O-Port 208, 214  
ICMP 311  
IDE-Controller 38  
if 274  
if else-Anweisung 270  
ifconfig 318  
IMAP 470  
inetd 337, 469  
I-Node 112  
I-Nodes 32  
inputrc 281  
insert module Siehe insmod  
insmod 178  
Installationsprozess 12  
Install-Skripts 192

Internet Super Daemon Siehe  
xinetd  
Interrupt 104, 208  
IP 303, 311, 328  
ip\_forward 405  
IP-Adresse 312, 313  
ipchains 401, 477  
ipchains-restore 411  
ipchains-save 411  
iptables 402  
IRQ 208  
ISA 207  
isapnp 214, 432  
ISO 304

---

## **J**

Jobnummer 101  
jobs 100  
join 121  
Journal 34

---

## **K**

KDE 242  
KeepAlive 471  
Kernel 169, 170  
Kernel konfigurieren 175  
Kernelparameter 211  
Kernel-Quellen 170  
Keyboard 234  
kill 106  
Koax-Kabel 307  
Kommandozeileneingabe 48

---

## **L**

LBA-Modus 417  
LD\_LIBRARY\_PATH 182, 423  
ldconfig 183  
ldd 182, 423  
LILO 143, 146, 421



---

lilo.conf 144  
line printer control Siehe lpc  
line printer daemon Siehe lpd  
link 110  
Linux Documentation Projects  
455  
Linux Loader Siehe LILO  
list 46  
ln *Siehe* link  
Logdateien 157  
Login-Shell 459  
logrotate 161, 460  
lokale Variable 20  
loopback Device 319  
lpc 253  
lpd 247  
lpr 255  
ls 24  
lsattr 85  
lsmod 176  
lspci 216  
LUN 220

---

## **M**

m4 Makro 367  
mail 370  
Mail Delivery Agent Siehe  
MDA  
Mail User Agent Siehe MUA  
mailq 371  
Mail-Server 363  
make 450  
make-directory 51  
Makefile 421  
man 89, 453  
Manpage 453  
MANPATH 91, 454  
Manual 91  
MASQ 404  
Masterbootrecord 143  
Maus 230  
MBR *Siehe* Masterbootrecord  
md5 394

MDA 363  
Message Transfer Agent *Siehe*  
MTA  
messages 160  
Metacharakter 25, 137  
Metatasten 13  
MIME-Typen 375  
mingetty 155  
mkdir 78  
mkfs 41  
modinfo 179  
modprobe 449, 450  
module info *Siehe* modinfo  
modules 176  
Motherboard 166, 224  
mount 60, 380, 391, 438  
move *Siehe* mv  
mozilla 99  
MTA 363  
mtab 63  
MUA 364, 470  
mv 108  
MX-Eintrag 370

---

## **N**

named 363  
Namensauflösungen *Siehe* dig  
NAT 312  
netstat 328, 466  
Network Time Protocol  
Daemon *Siehe* ntpd  
Netzmaske 315  
Netzwerk 303  
Netzwerkkarte 213  
Netzwerkklassen 312  
Netzwerktypen 310  
newaliases 369  
Newsgroup 93  
NFS 377, 473  
nfsstat 382  
nice 106  
nl 117  
nohup 106

---

nslookup 346  
NTFS 29, 299  
ntpd 165, 464  
number line 117

---

## **O**

Octetts 313  
OSI 304

---

## **P**

Pager 59  
Paketverwaltung 198  
PAM 396  
Partitionen 29  
Partitonieren 38  
Passive FTP 335  
passwd 71, 458  
Passwort 394  
Passwort-aging 74, 394  
paste 122  
PATH 27  
PC-Cards 222  
PCI 207, 216  
Peripheriegeräte 207  
PGP/GPG - Signatur 427  
PID 95  
ping 323, 324  
Pluggable Authentication  
    Modules Siehe PAM  
pnpdump 214, 432  
PointerDevice 230  
POP3 470  
Port 317  
Portnummer 333  
PostScript 250, 257  
pppd 468  
pr 258  
present working directory 50  
primary Name Server 359  
printcap 453  
printtool 253

Priorisieren Siehe renice  
proc 211, 217  
process status Siehe ps  
profile 27  
Programmbibliotheken 182  
Programmcode 280  
Prompt 19  
Protokolle 303, 311  
Prozess 104  
Prozessidentifikationsnummer  
    Siehe PID  
ps 95  
PS/2 230  
PS1 20  
PS2 21  
PS3 21  
PS4 21  
Pseudodevice 128  
pwconv 77, 458  
pwd 50

---

## **Q**

Quelltext 169  
quota 86, 89  
quotaoff 87  
quotaon 87  
quotastats 87

---

## **R**

RAID 219  
rc.sysinit 152  
RegEx 136, 436  
Reiser File System (RFS) 29  
Reiser-Filesystem 436  
rekursiv 80  
relativer Pfad 50  
remove Siehe rm  
remove module Siehe rmmod  
renice 107, 434  
repquota 88  
restore 300

---

Ring-Struktur 308  
rm 109, 441  
rmdir 110  
rmmod 177  
root 13  
route 316, 321  
rpc 377  
rpc.mountd 377  
rpc.nfsd 377  
rpm 193, 426  
rpm-Management 194  
RSA 416, 476  
Runlevel 149, 156  
Running 102

---

## S

Samba 377, 383  
Schleifen 274  
SCSI 218, 219  
SCSI-ID 218  
secondary Name Server 359  
Secure-Shell 411  
sed 136, 435  
sendmail 364  
set 19  
setpci 218  
setserial 208, 419  
SGID 84  
shadow 73  
shared Libraries 182  
Shell 18, 19, 137, 281  
Shell Scripts 267  
Shell-Funktionen 280  
shells 28  
Shell-Scripts 263  
Shellvariablen 19  
shutdown 155  
Sicherheit 393  
Sicherheitskopien 290  
SIGKILL 104  
Signal 104  
SIGTERM 106, 154  
Smart relay host 365

smb 442  
SMB 377  
smb.conf 383  
smbclient 387  
smbpasswd 387  
SOA 360  
Socket-Typ 469  
Softlink 110  
Sonderzeichen 156  
sort 125  
sound 215  
Sourcecode 169  
split 124  
Squashing 378, 473  
ssh 414  
sshd 412, 476  
ssh-keygen 416  
Standardfehlerkanal Siehe  
    stderr  
Standardkanäle 128  
Standardpasswort 400  
Standardport 317  
Start-Skripte 153  
statische Routen 316  
stderr 127  
STDERR 432  
stdin 128  
stdout 128  
STDOUT 432  
Sternverkabelung 307  
Stopped 102  
stream editor Siehe sed  
Subnetmask 314  
SUID-Bit 84  
Superblock 32, 41  
Superuser 50  
Swap-Partition 419  
Syslog 478  
syslogd 158, 159  
Systemvariablen 23

---

## T

tac 116

---

tail 115, 460  
tape archiv Siehe tar  
tar 171, 172, 187, 292, 295  
TCP 303, 311  
TCP/IP 304, 328  
tcpd 337  
tcpdump 331, 467  
TCP-Wrapper 393  
tee 133, 432  
telinit 154  
telnet 343  
Telnet-Server 344  
Terminal 17, 100  
Terminalprogramm 19  
Terminierung 220  
testparm 385  
Tests 113  
Texte bearbeiten 115  
Thin Ethernet 307  
Token Ring 309  
top 96  
touch 52  
tr 134  
traceroute 326  
translate Siehe tr  
TTL 360, 471  
tune2fs 44

---

## **U**

Übertragungsverfahren 310  
UDP 303, 311  
UID 77  
ulimit 399  
umask 83, 441  
Umgebungsvariablen 19  
umount 62  
unalias 49  
uname 173  
unexpand 123  
uniq 131  
Unix-Streams 328  
until do-Anweisung 278  
USB 207, 221

usbmodules 222  
useradd 67, 457  
userdel 75  
usermod 76

---

## **V**

Variablen exportieren 23  
Verzeichnisstruktur 35  
vi 53, 436  
video mode tuner 227  
visual editor 53  
VLB 207  
Vordergrund Siehe fg

---

## **W**

wc 116, 430  
webmin 424  
Web-Server 371  
whatis 92  
while do-Anweisung 276  
whois 353  
Winmodem 417  
word count Siehe wc  
wvdial 468

---

## **X**

X Display-Manager Siehe xdm  
X11 223  
xargs 140, 432  
xdm 243  
xf86cfg 227  
xf86config 229  
XF86Config 226  
XFree86 223, 227, 233  
xinetd 337, 469  
xinit 241  
xinitrc 241  
Xinstall.sh 224

---

X-Server 227, 234, 241, 243,  
246  
X-Servers 443, 447  
xsession 244  
xset 444  
X-Terminal 446  
xvidtune 227  
X-Windows 15, 240

---

## **Z**

zcat 185  
Zeitserver Siehe ntpd  
zImage 176  
zoneinfo 165

## Mit Bestsellern aus dem Bereich IT lernen

Dietmar Abts

### **Grundkurs JAVA**

Von den Grundlagen bis zu Datenbank- und Netzanwendungen

4., verb. u. erw. Aufl. 2004. X, 408 S. mit Online-Service. Br. € 19,90

ISBN 3-528-35711-8

Klassen, Objekte, Interfaces und Pakete - Ein- und Ausgabe - Multithreading  
- Grafische Oberflächen mit Swing, Applets - Datenbankzugriffe mit JDBC -  
Netzanwendungen - Spracherweiterungen der Version J2SE 5.0

André Maassen/Markus Schoenen/Ina Werr

### **Grundkurs SAP R/3®**

Lern- und Arbeitsbuch mit durchgehendem Fallbeispiel -

Konzepte, Vorgehensweisen und Zusammenhänge mit Geschäftsprozessen

3., durchges. u. verb. Aufl. 2005. XXIV, 608 S. mit 256 Abb. u. 25 Tab.

Br. € 39,90

ISBN 3-528-25790-3

Technische Aspekte - Benutzerkonzept und Handhabung - Unternehmensstrukturen in Personalwirtschaft, Materialwirtschaft, Vertrieb und Finanzwesen - Fallstudiengestützte Einführung in die Arbeit mit Stammdaten und Bewegungsdaten - Screenshot-basierte Arbeitsanweisungen - Erste Schritte mit Report Painter und Report Writer

Dietrich May

### **Grundkurs Software-Entwicklung mit C++**

Eine praxisorientierte Einführung - Mit zahlreichen Beispielen, Aufgaben und Tipps zum Lernen und Nachschlagen

2003. XVI, 532 S. Br. € 29,90

ISBN 3-528-05859-5



Abraham-Lincoln-Straße 46  
65189 Wiesbaden  
Fax 0611.7878-400  
www.vieweg-it.de

Stand 1.7.2005. Änderungen vorbehalten.  
Erhältlich im Buchhandel oder im Verlag.